(2)

DTIC FILE COPY

HSD-TP-88-002

DETAILED DESIGN SPECIFICATION FOR A PROTOTYPE ASSESSMENT SYSTEM FOR AIRCRAFT NOISE (ASAN)

Sanford Fidell
Michael Harris
Nicholaas Reddinguis

BBN Laboratories Incorporated
21120 Vanowen Street
P.O. Box 633
Canoga Park, CA 91303

July 1988

Interim Report for Period February 1987 - October 1987

Approved for public release; distribution is unlimited.

Noise and Sonic Boom Impact Technology Program
Systems Acquisition Division
Human Systems Division
Brooks Air Force Base, TX 78235-5000

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS: None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): BBN Report No. 6499	5. MONITORING ORGANIZATION REPORT NUMBER(S): HSD-TP-88-002
6a. NAME OF PERFORMING ORGANIZATION: BBN Laboratories Incorporated; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Human Systems Division, Noise and Sonic Boom Impact Technology (NSBIT)
6c. ADDRESS (City, State, and ZIP Code): 21120 Vanowen Street, P.O. Box 633, Canoga Park, CA 91303	7b. ADDRESS (City, State, and ZIP Code): OL-AC HSD/YA-NSBIT, Wright Patterson AFB, OH 45433-6573
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Noise and Sonic Boom Impact Technology Program; 8b. OFFICE SYMBOL (If applicable): HSD/YA-NSBIT	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F33615-86-C-0530
8c. ADDRESS (City, State, and ZIP Code): OL-AC HSD/YA-NSBIT, Wright Patterson AFB, OH 45433-6573	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. 63723F; PROJECT NO. 3037; TASK NO. 0003; WORK UNIT ACCESSION NO. 02

11. TITLE (Include Security Classification): Detailed Design Specification for a Prototype Assessment System for Aircraft Noise (ASAN) (Unclassified)

12. PERSONAL AUTHOR(S): Fidell, Sanford; Harris, Michael; Reddingius, Nicolaas

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim	FROM 2/12/87 TO 10/15/87	1988 July	93

16. SUPPLEMENTARY NOTATION

17: COSATI CODES		
FIELD	GROUP	SUB-GROUP
12	07	
24	02	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): Computer Systems, Computer Software, Noise, Aircraft Noise, Noise Effects, Noise Impacts, Animals, Humans, Structures, Models, Noise Prediction, SDev

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The U.S. Air Force Noise and Sonic Boom Impact Technology (NSBIT) Program is sponsoring a multi-stage effort to create a computer system containing tools needed by the environmental planning community to perform a variety of tasks related to assessing the environmental impacts of aircraft noise on people, animals, and structures. This interim report provides a detailed design specification for a prototype version of the NSBIT Assessment System for Aircraft Noise (ASAN) that is the major product of the first stage of this effort.

The purposes and expected uses of ASAN are presented in Fidell and Harris (1987). The general functional capabilities of this system are described by Harris and Fidell (1987). The current report describes the organization of ASAN, its functional capabilities, and its major software modules.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFED
22a NAME OF RESPONSIBLE INDIVIDUAL: Lt Col Geral Long	22b. TELEPHONE (Include Area Code): 513-255-8416/8417; 22c. OFFICE SYMBOL: HSD/YA-NSBIT

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

Summary

This report presents a detailed design specification for a prototype version of the Assessment System of Aircraft Noise (ASAN). It is the major product of the first stage of a multi-year effort, sponsored by the U.S. Air Force Noise and Sonic Boom Impact Technology Program, to create the computer-based tools necessary to the environmental planning community for the performance of a variety of tasks related to assessing the environmental impacts of aircraft noise effects on humans, animals, and physical structures.

The goals, design philosophy and intended users of ASAN are presented in Fidell and Harris (1987). The general functional capabilities of the system are described by Harris and Fidell (1987). Excerpts from these and other documents produced under Task Order 0003 of Contract F33615-86-C-0530 are included in this report for the convenience of the reader.

The organization of ASAN, its functional capabilities and its major software modules are described in this report.

Preface

This report was prepared under Contract F33615-86-C-0530 of the Noise and Sonic Boom Impact Technology (NSBIT) Program. The NSBIT Program is conducted by the United States Air Force Systems Command, Human Systems Division under the direction of Lt Col Geral Long, Program Manager.

The work herein is in fulfillment of Subtask 3.4 of Task Order 0003.

Acknowledgments

The authors acknowledge the advice received from Air Force personnel in the course of preparing this report. Special appreciation is extended to Mr. Lawrence S. Finegold, Mr. Scott Hall and Lt Col Geral Long for their suggestions concerning the conduct of this effort and for their comments

Table of Contents

List of Figures

1. Introduction

This report provides a detailed design specification for an environmental planning aid system known as ASAN (for Assessment System for Aircraft Noise). The system is intended for use by members of the Air Force environmental planning community in conducting the noise-related portions of environmental impact assessments for Military Operating Areas (MOAs) and Military Training Routes (MTRs).

This specification is intended to guide both short and long term development of ASAN, as required by section 3.4 of the statement of work of Contract F33615-86-C-0530. "Short term" refers to a prototype version of ASAN to be demonstrated early in calendar year 1988. Only a subset of the capability described in this report will be implemented for demonstration purposes. As with any prototype system, it is expected that these specifications will be amended on the basis of operational experience.

This document specifies the software component of the ASAN system. Appendix A outlines the reasoning which led to selection of the Zenith 248 personal computer as the host for the prototype version of ASAN. Appendix B describes the hardware configuration for the prototype system.

The primary user for whom ASAN is intended is an Air Force officer or civilian serving in a major command or environmental planning office. This end user is expected to have only rudimentary computing skills, limited access to a computer larger than a desktop machine, and only a basic understanding of environmental acoustics. Although ASAN contains a number of features which may be exploited by more sophisticated users, the user interface and general mode of operation of the software are designed for inexperienced users.

The following were among the principles adopted in developing the overall design of ASAN:

- Software should facilitate the basic aspects (e.g., text, graphics, and document handling) of the environmental planner's job to the greatest extent possible.
- Software tools should not be limited to those which implement the way that environmental planners currently perform their jobs, but instead should provide environmental planners with tools that make it possible for them to work more effectively and productively.
- Software should be modular (producible and operable in parts) and extendible as time and budget permit, so that it can eventually provide assistance in all aspects of the environmental planner's job.
- Software should be organized in such a manner that it can take maximal advantage of existing programs and databases.
- Software should not be developed from scratch if it can obtained by modifying existing code, and it should not be developed at all if the desired capability can be purchased at reasonable expense.
- As much of the software as possible should be executable on personal computers readily available to environmental planners at the base level.

- The software system should permit the burden of updating and maintaining the databases on which it relies to be shared in an efficient manner between a central responsible organization and local environmental planning personnel.

2. General System Objectives

The objectives for ASAN are to provide a set of tools to assist in predicting and assessing environmental impacts due to noise associated with USAF flight operations. Three types of tools are to be provided:

1. Process oriented tools, which assist in:
    - Determining the steps required to complete an environmental assessment and tracking the status of the work as it progresses through its stages.
    - Finding likely information sources for the unstructured parts of the data gathering and analysis phases.
    - Handling the "fixed" data (e.g., maps, missions, aircraft noise characteristics) in an organized and effective manner.
2. Analytic tools, which assist in:
    - Performing calculations to predict noise exposure from proposed operations based on current best engineering practice.
    - Making assessments of the impacts of these predictions based on current knowledge of the effects of noise exposure on the environment.
    - Reporting the engineering and assessment results in a form useful for incorporation into an Environmental Impact Statement (EIS), Environmental Assessment (EA) or Finding of No Significant Impact (FONSI).
3. Organizational tools, which assist in:
    - Documenting analytic work and decisions for review by cognizant authority.
    - Keeping records required for compliance with the National Environmental Policy Act (NEPA).
    - Keeping supporting information accessible and understandable to successors after an environmental planner has been reassigned.

The primary objective of ASAN is to implement the state of the art in environmental assessment. ASAN's design must make provision, however, for incorporating improvements in the state of the art as they become acceptable practice during the system's life cycle.

2.1 User's Perspective of ASAN

Figure 2-1 is a block diagram of the functional capability of ASAN from the user's point of view. The first phase of the environmental assessment process is inevitably problem definition. Users must supply the system with at least fragmentary information about the noise sources and geographic areas of concern. As more detailed information about mission requirements, land uses, and the like becomes available, users continue to interact with the software modules that accept geographically-oriented problem definition data. This iterative problem definition phase is expected to be convenient enough that users will eventually rely on computer based methods for data entry and retrieval to organize their routine handling of documents used in the environmental assessment process.

The primary mode of operation of ASAN is intended to be interactive and iterative in real time. Since an environmental planner is unlikely to have at hand all of the information or all of the time necessary to complete an environmental assessment in a single computing session, several provisions are made in the organization of the system to permit users to enter and manipulate even small amounts of information efficiently. The software is also organized to permit environmental planners to consider several alternative proposed actions at a time, either as variants of the same action or as independent projects.

End users will acquire some of the relevant data, *such as terrain maps, land use maps and aircraft* noise emission levels and contours, from centralized sources. Other data, such as proposed routes, locations of residential areas, and point of contact lists, may be updated locally. The product of the problem definition phase is a geographically-organized composite data structure, with many superimposed "layers" of information.

The next step in conducting an environmental assessment is analytic. The initial part of the analysis generates estimates of the noise exposure created by flight activity for specified points or areas. Exposure estimation is also expected to be an iterative process in which users may compare in increasing detail the relative contributions of a variety of noise sources to total exposure. All such estimates are stored in geodatabases for convenience of manipulation and interim display.

The final portion of the analytic process is estimation of effects. A number of analytic procedures, implemented as independent software modules, operate on the layered composite data structure. The ultimate product of this phase is a set of updated databases available to the next phase, report generation.

In the report generation phase, the collected data and analysis results are combined either automatically or under user control to produce text and graphic material for interim and final reports. The data and displays produced during the analysis phase are used to generate documents containing both prose and graphics. A sample of the sort of text that ASAN's report generator will eventually be able to produce may be found in Appendix C.

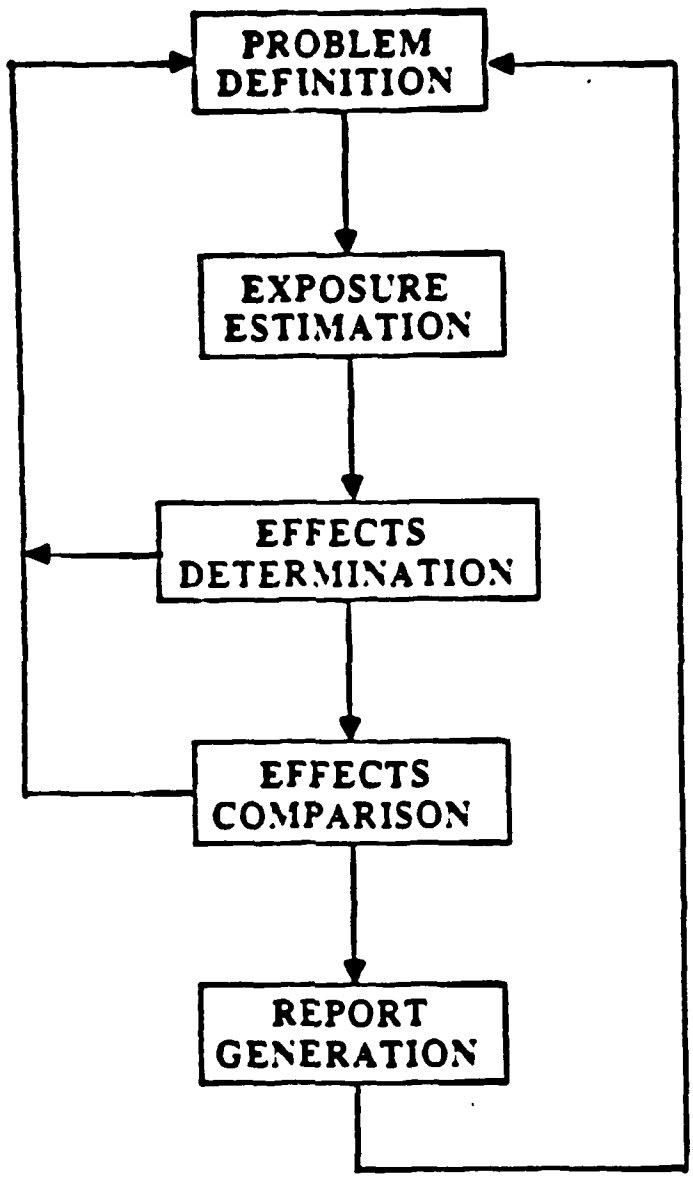

Figure 2-1: ASAN from the User's Perspective

2.2 Major Features of ASAN

The major features of ASAN are discussed in the following four sections. The first of these describes ASAN's software infrastructure, the internal organization of the program with which the user ordinarily has little contact. The remaining three subsections describe the features and capabilities that users will encounter in the problem definition, analysis, and report generation phases of ASAN use.

2.3 Software Infrastructure

Some of the features of ASAN which are essentially invisible to end users are among those which are the most important for achieving increased accuracy and efficiency in conducting environmental assessments by automated means. These are addressed below.

2.3.1 Checklist Directed Activity

ASAN includes a set of checklists itemizing the mandatory and optional steps for completing an environmental assessment. A software module keeps track of progress by reference to the appropriate checklists. Thus, ASAN provides a means for the environmental planner to know, at any point in the process, which tasks have been completed and which remain to be undertaken.

2.3.2 Audit Trails and Backup

Relevant details about every operation performed on ASAN are automatically recorded in an audit trail, providing a complete activity record. The audit trail would include, for example, when a data item was last updated, by whom, and in what way. One software module records the current state while another module restores the data structures and processing activities to a previous state. This mechanism forms the basis of a reliable backup capability, as well as the means for undoing actions if necessary.

2.3.3 Tutorial Mode

Provision is also made for a form of user-invocable embedded training. Although it is not planned to implement this facility in the prototype version, a complete introduction to system capabilities will eventually permit self-instruction by end users. This capability is intended to supplement rather than replace on-line help.

2.3.4 Database Managers

The design of ASAN is data driven. The system is built on a set of databases that represent both geographic and conventional (i.e., text or numeric) data. Throughout this document the term "geodata" is used for information that is inherently tied to representation by means of maps. Other data is referred to as "text data."

The fine points of this distinction and the details of managing the databases that support ASAN are not of direct concern to the user and are incorporated in the data management portion of the control logic. For reasons summarized in Appendix D, the GRASS Geodata Information System was selected as the geodata manager, and ORACLE was chosen as the text database management system for the demonstration version.

2.4 Problem Definition Phase

The software modules that implement the Problem Definition Phase of ASAN are those with which users who are performing environmental impact assessments will probably interact first and most frequently. They are organized around a set of databases listed in Appendix E to permit users to store and recall a wide range of geographically-oriented information. These databases provide a means for preserving the daily progress made by environmental planners in identifying noise impacts. Continuous updating of these databases permits planners to conduct their environmental analyses incrementally, whenever sufficient new information is available to warrant a detailed evaluation.

2.4.1 Centrally Maintained Databases

Many of ASAN's databases, including most of those which contain geographic information (geodata), are essentially permanent in nature and will require only infrequent revision. All semi-permanent databases should be created and maintained by a central organization such as the Air Force Engineering and Services Center at Tyndall Air Force Base (AFESC). AFESC (or another organization either within or outside of the Air Force) would acquire and validate raw information, combine and pre-process it as necessary to make it directly usable within ASAN, and distribute it on suitable media. Similar considerations apply to non-cartographic databases (e.g., aircraft noise characteristics).

2.4.2 Locally Maintained Databases

Provision will be made in ASAN to accommodate a variety of cartographic and descriptive materials which end users develop locally to conduct environmental assessments. The ASAN software will enable the user to construct and maintain this locally generated information together with the semi-permanent databases in a manner that preserves data integrity while minimizing storage requirements.

2.4.3 Information Hygiene

Since some of the information that end users may enter into ASAN is not readily available in digital form, or may not be reliable, a process of data entry and verification is necessary. Furthermore, considerable effort and computational resources may be required to transform some of the geodata to allow combinatorial viewing and analysis: mathematical conversions and alterations may be necessary to units of measure, angular relationships, scales, resolutions, and projection methods. The bulk of this data collection and processing need not concern end users, however, since it is intended that such efforts be performed at a central facility.

The problem of quality control of information used in environmental assessments, either as currently performed or as they may be performed in the future with computer assistance, is a continual concern. While ASAN will facilitate central management of release levels of global data sets, no run time management function to assure the quality of locally generated cartographic information is anticipated. The only indirect form of quality control inherent in ASAN is the ability for headquarters personnel to review the work of end users in a convenient and complete digital form.

Both database systems used by ASAN are available as a set of library routines callable from an application program. It is in this form that they will be incorporated into ASAN. Both systems are also capable of operation in stand-alone mode. While stand-alone operation would enable users to perform ad hoc queries, there are good reasons for denying end users direct access to ASAN's databases:

- The internal representation of the various ASAN databases are not necessarily intuitive to the user, limiting the usefulness of direct queries.
- ASAN must maintain information about the state of its databases. Providing users with direct access to the databases would permit undocumented modifications of them, and an attendant loss of auditability.

In fact, one of the main goals of the ASAN system is to shield the user from the complexity of direct manipulation of source data through a set of user interface screens and sophisticated audits and checklists. Providing the tools to the user to circumvent these safeguards would defeat this goal.

2.5 Analysis Phase

The analysis phase can be undertaken whenever a user has provided enough information to define a problem at a useful level of detail. It is expected that most analyses required for environmental assessments will be performed several times at increasing levels of detail during the course of evaluation of a proposed action. Users will probably wish to perform an initial quicklook set of analyses as soon as the outlines of a proposed action are known. As users acquire and enter greater amounts of detail about a proposed action they will want to perform more refined analyses. The design of the analysis phase of ASAN is intended to facilitate this sort of iterative use. The analysis phase is accomplished in two steps, as described below.

2.5.1 Noise Exposure Prediction

The first step estimates the noise exposure produced by flight operations in appropriate units for the various noise effects prediction modules. For example, noise exposure will be estimated in units of L_{dn} for purposes of predicting the prevalence of annoyance, but in units of sonic boom overpressure for prediction of structural damage. More detailed predictions of spectral shapes and time histories are needed for purposes of predicting speech interference effects of aircraft noise exposure. The output from these modules will typically be in the form of a set of database entries which the user may view as a map layer representing the physical aspects of the environment created by flight operations.

2.5.2 Noise Effects Estimation

The second step of the analysis phase invokes a set of noise effects prediction modules to operate on the exposure estimates developed in the first step. The outputs of these modules are sets of data blocks that contain numeric codes representing predictions of noise impacts on the environment. Effects are classified as to the severity or their impact (from none to very severe) on a scale appropriate to the effect.

Output data blocks produced by the noise effects predictions modules are treated in two ways. First, they are used by the report generation module for expansion into a text fragment, e.g., "Noise exposure in the area may be expected to be incompatible with single family residential land use." Alternatively, the data can be shown pictorially as a map layer, e.g., a map showing all areas incompatible with single family residential development.

2.6 Report Generation Phase

The data blocks produced during the analysis phase are manipulated in the report generation phase to produce prose text for inclusion in draft or final documents. Text and graphics outputs can be generated either by user direction or automatically from templates.

2.6.1 Effects Prose Generation

The information generated by the analytic modules is in the form of a group of global data blocks, each containing a keyword text string and a list of numerical or textual metrics. The Effects Prose Generator uses this information to access the Noise Exposure and Effects Boilerplate Text Database ("Boilerplate Database") to produce prose descriptions of the effects (c.f. Appendix C).

The Boilerplate Database is accessible by index on effect and severity so that records can be related to effects assessment. These records include fragments of prose text containing "placeholders" (data format descriptors). For each analysis output data block, the Effects Prose Generator searches the Boilerplate Database for matching keywords. The placeholders in the associated prose fragments are replaced by the metrics from the data block, and the completed prose fragment is stored.

2.6.2 Prose Optimization

When arranged in the sequence preferred by the planner, the prose fragments produced by the Prose Generator may appear redundant or stilted. Furthermore, the text created by simple chaining of all available text fragments may not create the proper overall impression of the environmental impacts associated with a proposed action.

The Prose Optimizer module reorganizes and rearranges text fragments into less redundant and more understandable prose, by applying a set of applicable rules of style. This module will not be included in the demonstration prototype system, but will be developed later in the project.

2.6.3 Word Processor

An already developed word processing facility will be chosen and integrated into ASAN. Its purpose is to allow the planner to rearrange and edit the text materials in databases or generated reports. This module will not be included in the demonstration prototype system, but will be developed later in the project.

2.6.4 Document Formatter

An already developed, device independent document formatting facility will be integrated into ASAN. This module will not be included in the demonstration system, but will be developed later in the project.

2.6.5 Hardcopy Output Management

This module directs text and graphic hardcopy output to a specific device selected by the user at run time. It supports a variety of graphic display and hard copy hardware, formatting and transmitting the data in a manner appropriate for each. It implements spooling to pass graphic outputs to the selected device without tying up the host processor, to permit productive use of the environmental planner's time while output is in progress.

3. Software Design Issues

ASAN is intended to be an integrated software system that provides environmental planners with access to the best current engineering practice. This implies that ASAN must provide access not only to specialized knowledge about diverse environmental effects of noise exposure, but also with access to a variety of calculation-intensive analyses. In the face of this complexity, a major goal of the software development effort is to avoid presenting end users with a disjointed set of tools that only experts can effectively use. Development of ASAN will be firmly grounded in modern software system design theory to avoid this outcome. As a consequence, ASAN will consist of four largely independent structures:

1. Databases - generic and job specific information
2. User Interface - dialogue between the user and the system
3. Analytic Models - analysis and engineering "tools"
4. Control Logic - the system software that exercises all other modules

Basic concepts of system design theory, including modularity and independence of data and control structures, and their implications for ASAN are described in Appendix D.

3.1 Implementation Environment

The implementation language chosen for ASAN is the "C" language. C is a very standardized high-level structured language. It is highly portable: that is, a C program will, with very little if any modification, run on a very large number of computers. It is a language that enforces a certain amount of structure, while at the same time providing access to hardware features at a level usually reserved for assembly language programs. Excellent software development tools for C are available, including a fully-featured compiler and a source level debugger. For these reasons it has become the language of choice for many major systems development projects.

3.1.1 Initial Demonstration System

The prototype version of ASAN is intended to run in the standard hardware and operating system environment of the Air Force's Zenith 248 personal computers, configured as described in Appendix B. When alternative implementation options are available, consideration will be given to software portability to functionally similar hardware.

3.1.2 Expected Developments in 80X86-based Microcomputer Systems

Several important developments in microcomputer technology are expected in the next few years. 32-bit processor chips that are significantly faster than the Z-248's Intel 80286 processor (and the circuit boards that can keep up with them) are currently entering the market at reasonable prices. Operating systems are now under development for computers with 80286 and 80386 CPUs which will enable protected mode operations. These operating systems will probably be released within the next calendar year, and will permit two major improvements in the operating environment for ASAN:

- Access by application programs to more than 640 kB RAM, the current limit imposed by the MS-DOS operating system.
- The appearance of multi-tasking operating systems, which allow several processes to operate in the computer simultaneously.

By the time the NSBIT program is concluded in 1992 and ASAN is transitioned to a maintenance organization, the combination of these two factors will provide current minicomputer performance capabilities in desktop personal computers. Such performance levels will allow at least some highly compute-intensive analyses to be performed in a background mode on desktop machines while the environmental planner performs other tasks interactively in the foreground.

3.1.3 Operational Environment

The operational version of ASAN will be distributed at a time when the marketplace will provide considerably more capable desktop machines. It is likely that such machines will eventually become available to Air Force environmental planners. ASAN will be designed so that later releases will take advantage of this enhanced capability.

The demonstration version of ASAN will show how the operational system is intended to function, and will provide a vehicle to assess the benefits that could accrue from technology beyond the Zenith Z-248 computer system.

3.2 Data Management in ASAN

ASAN will be built on two data managers: one to handle geographically oriented data ("geodata") and one to handle text data, i.e., text strings associated with the maps and all other data that do not have latitude and longitude associated with them. From a pure data management perspective there is no difference in principle between these two types of data. The decision to use two systems is motivated by practical considerations.

The objective of ASAN is to deliver an environmental planning tool, not a new data manager. The functions needed to accommodate the unique problems of managing geodata (e.g., transformation to accommodate different projections and scale) are already incorporated in the geodata manager (GRASS). This specialized tool does not, however, provide the capability to manage text data.

3.2.1 Geodata Manager

The Geodata Manager module performs all maintenance and access operations on geographically organized data, using the graphic display subsystem for geodata displays and the console display screen for control displays. This task is complex and compute-intensive, but is well understood, and several existing software packages possess the necessary functionality and quality.

The Geodata Manager will be able to:

- manipulate databases containing mixed text and graphic data.
- specify and manage color in a logical and efficient manner.
- carry out geodata entry and maintenance (addition, deletion, modification) in an efficient manner.
- specify points in the data space in spatial or verbal relational terms, e.g., "all residential areas within 10 miles of a flight track."
- locate a point or area in a geodatabase rapidly.
- perform "join" operations, in which data from several separate geodatabases can be manipulated as though contained in a single database.
- combine geodata to produce composite data structures and displays.
- produce well composed and formatted graphic output, in color, on a variety of display and hard copy devices.
- operate according to predefined procedures written in a simple dialect.
- implement all capabilities as library functions callable from external C language programs.
- use system resources efficiently, especially the mass storage and graphic display hardware subsystems.

- impose no significant limitations on the maximum size or geographic area of a database.
- perform display operations in a device independent manner, so that graphics display subsystems can be interchanged with minimal changes to the software.
- carry out command operations from graphics input devices in a device independent manner, so that one graphics input device can be substituted for another, or so that operations can continue without any graphics input device.

The preferred geodata manager for the prototype system is GRASS, for reasons described in greater detail in Appendix D.

3.2.2 Text Data Manager

The Text Data Manager software performs all management operations for text information. These operations will be initiated either by the analytical modules in ASAN or by the operator using the console display screen. This task is a straightforward one that should not tax the capabilities of most database management systems. The Text Data Manager will be able to:

- carry out text data entry, addition, deletion, and modification in a simple manner.
- locate data records rapidly and efficiently.
- perform "join" operations, in which data from separate data sets can be manipulated as though contained in a single data set.
- operate according to predefined programs or "macros" written in an understandable dialect.
- implement all capabilities as a library of functions callable from external C language programs.
- use system resources efficiently, especially mass storage.
- impose no serious limitations on the maximum size of a database.

The Text Data Manager selected for the prototype version of ASAN is ORACLE, for reasons discussed in Appendix D.

3.3 Modularity

As has been pointed out above, the nature of the analyses and processes to be performed by ASAN is such that a high degree of modularity in the program logic is required. It must be a simple matter to replace a calculation module with a new version or add a new module to the system when needed. This results in the following three design criteria.

3.3.1 Software Independence

The program logic itself must be modular. To the maximum extent possible, code shared among modules must be available to all modules as a library. This will insure consistency of approach throughout the system as well as minimize cost of development and maintenance.

There must not be any logic links between different modules. All modules will communicate only through files in the database. This will insure that no module will be dependent on the internal structure of another module, so that changes in program logic remain local to the module being changed.

3.3.2 Data Independence

No analytic modules will access the database directly; they will instead go through the ASAN database management system (part of the ASAN control modules). This is a necessary requirement to maintain an audit trail. It also enables error correction, and allows the physical location and structure of the database to be changed with minimal impact on the application code.

To achieve a certain amount of data independence, the ASAN database management system will have access to a data dictionary. The data dictionary is used to locate the information in the database, to validate the data and to maintain a record of the age of the information.

The data manager must also have access to a dependency chart, which shows how derived information (i.e. theoretically redundant data) is related to primary information basic (i.e., the normalized set). This is required to insure that derived and/or duplicated information remains synchronized with the primary information. This is a design compromise for performance reasons: while a properly structured normalized database will not contain derivable information, the amount of calculation required to (re-)generate it on demand becomes prohibitive in many practical situations.

3.3.3 Configuration Management

The ASAN system is composed of many independent modules operating on and communicating through independent data structures. In this environment, where system components can be replaced or modified as individual units, compatibility is a major concern. This issue is addressed by the inclusion in ASAN of a configuration database and associated validation software.

Every software module and database schema (structure description) has an embedded revision level code. The configuration database contains a list of all software modules and database schemas necessary for system operation, and gives the correct revision level code for each. At system startup, and at other appropriate times during system operation, the validation software checks the actual component revision

level codes with those given in the configuration database; if a "skew" condition is found to exist, it is reported to the user and system activity is suspended until it is resolved.

3.4 Control Flow Diagram

Figure 3-1 is a block diagram of the major parts of ASAN. The full capability shown in the figure will be constructed in stages. In the first stage of the development effort, the control structures and top level control screens will be implemented, but all other software components will be represented by code stubs. In succeeding stages, the stubs at the next lower levels of the design hierarchy will be expanded into functional software modules. This approach is recommended not only for efficiency in preparing code, but also because noise exposure prediction and other code being developed under the sponsorship of NSBIT and other Air Force organizations are not immediately available for incorporation into the system.

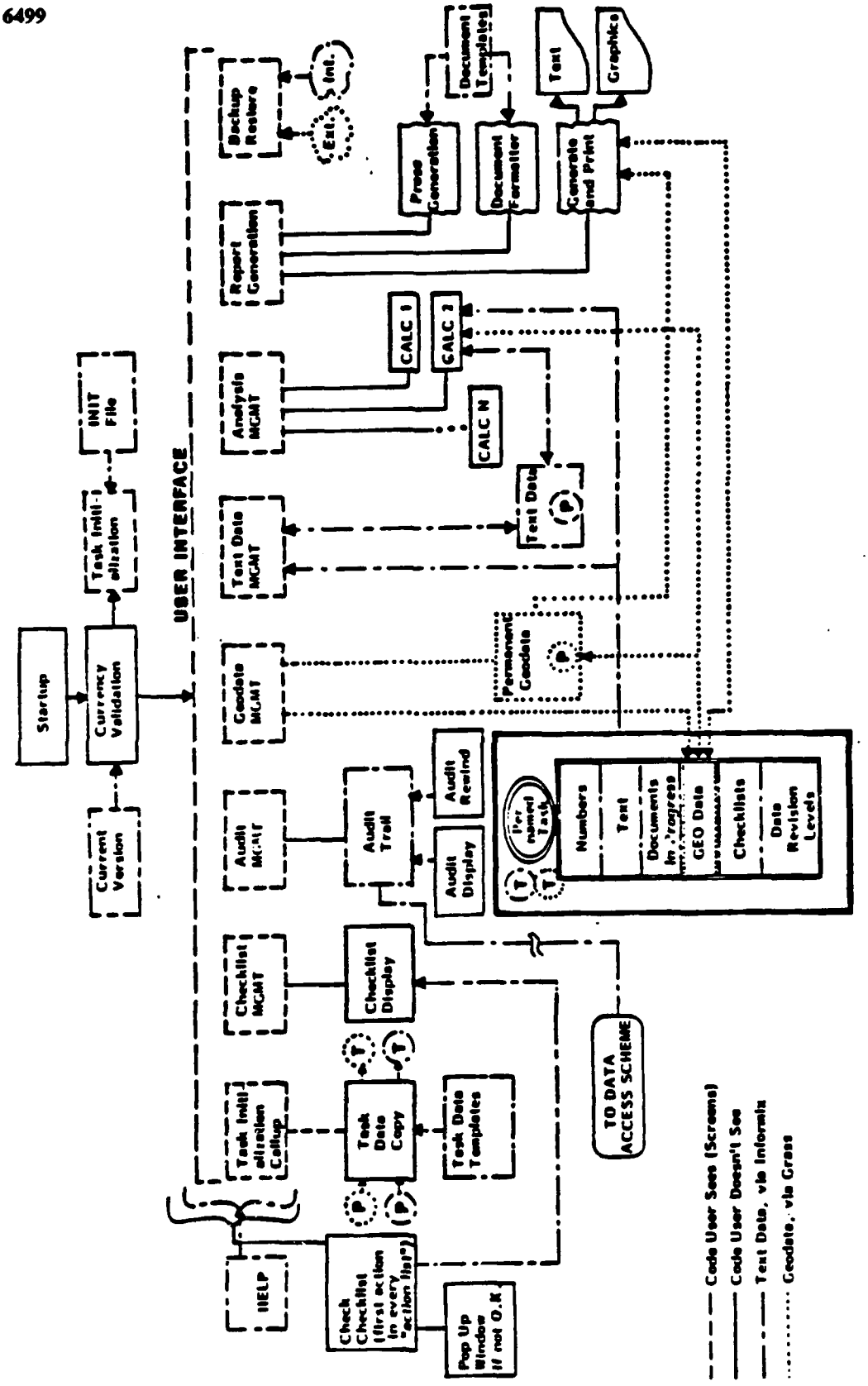

Figure 3-1: ASAN Block Diagram

4. Data Management and Data Integrity

Almost all activities of the environmental planner involve the manipulation of multiple sets of information. From the system's perspective, maps, analyses and checklists are all treated as user views of certain portions of the internal databases. It is therefore important to describe ASAN design measures intended to aid environmental planners in database manipulations. As noted above, the prototype ASAN data management system is built on top of two data managers: GRASS and ORACLE. This chapter discusses the data management function of ASAN'S control logic.

4.1 Data Access Controls

The design of ASAN is predicated upon modern database management techniques, as discussed in Appendix D. ASAN's control logic builds on the primitives supplied by the text and geodata managers (GRASS and ORACLE) to implement as many of these techniques as are appropriate. Certain complexities are unavoidable because, in addition to the usual data storage and retrieval operations, ASAN must also keep track of the update levels and computational relationships among various files. ASAN is similar to other sophisticated applications in this respect; the basic functions of database management systems are the primitives upon which a more elaborate data interface is built.

In order to maintain the integrity of ASAN's databases, the data management routines are the only procedures that access disk files. The ORACLE functions AUDIT, COMMIT WORK, etc. are used to control the integrity of the database. To insure integrity of the data, these functions are centralized in the set of C-callable ASAN data management routines which perform the following functions:

READ (Simple) The data manager will read the information requested.

READ (for Calculation) The data manager will read the (derived) information requested and check the data dependency table to insure that it is valid.

MODIFY The data manager will read the original data record from the database and write it to the audit file, then write the new record.

DELETE The data manager will read the original data record from the database and write it to the audit file before deleting the record.

ADD The data manager will write a "null" record to the audit file before appending the new record.

4.2 Data Administration

Data administration is the function that determines what information is available to ASAN, what update levels are centrally supported, what types of data files are supported and how they are organized. In the long term, the function will become part of the on-going configuration control and maintenance of ASAN. As more modules are developed for ASAN and as new analyses are included, corresponding updates to the configuration and module currency databases must be made by a support organization such as AFESC.

A data administrator is the custodian of the information. This person maintains the data dictionary, the data dependency file and access control modules. This function should be performed at a centralized location such as AFESC. As with any custodial or audit function, it should be separate from those (centralized) functions that work with the data.

Ideally, a close but adversarial relationship should exist between the data administrator and the users and providers of data. Before new databases or individual pieces of information can be added to ASAN, the provider (or user) of data must convince the data administrator that the information is neither available nor derivable from information already available within ASAN. Alternatively, the data administrator must be convinced of the need to make a structural change and thereby trigger the corresponding release controls before a change in is made in ASAN's data structures.

4.3 Database Administration

The database administration function is charged with the actual day-to-day maintenance of the database. It is a purely technical function concerned with the efficient physical implementation of the database. This function should also coordinate the acquisition of central databases and release updates to the user community. The function also maintains the test databases used by maintenance and development programmers.

4.4 Data Integrity, Logs and Audits

ASAN's data management system will allow various levels of access to users of the system. Certain types of information cannot be modified by the user (e.g., effects criteria, standard aircraft data). Other types of information will be created and updated as part of the assessment process.

ASAN will maintain data integrity through audit files, which record changes made to the database in such a way that a record is kept of how the database evolved from its original to its present state. When complex database manipulations take place, involving many records in many data tables, the system will create a transaction log. That is, all modifications to the data tables are suspended until it has been ascertained that all updates and calculations have been completed without error before the modifications are committed and the database updated.

Transaction logs and audit files have additional uses beyond their function in the data management system to insure that data integrity is maintained. They also serve as the basis for a user reset function to allow several scenarios to be started from a common starting point in the analysis. Transaction logs and audit trails also provide (after considerable condensation) the basis for compliance with the NEPA requirement for a complete record of decision.

5. Program Structure

The ASAN software system consists of four subsystems:

1. User Interface and Control Subsystem
2. Data Management Subsystem
3. Analytic Subsystem
4. Report Generation Subsystem

In this Chapter, the ASAN system control structure and software interfaces are described, and the flow of control is illustrated. Other software modules are discussed in a succeeding Chapter.

5.1 Control Subsystem

The control subsystem invokes, directs and supervises the activities of all the other ASAN modules. It is implemented as a screen-oriented user interface package and a set of screen descriptions which drive the interface. The user commands the operation of ASAN through a series of screen displays, selecting, activating or filling in displayed items using the console pointing device (mouse) and/or standard keyboard.

5.1.1 User Interface Package

The screen-oriented user interface recommended for demonstrating the prototype version of ASAN is a BBN-proprietary package. The package is written in C for Unix and Unix-like (Xenix) operating environments, and presently supports monochrome terminals which use the ANSI standard terminal control protocols.

For this application, the user interface package will be modified to run in the MS-DOS environment, its current dependency on a system-resident screen control library will be eliminated, and support for the EGA-equivalent color console display will be added. Additional information about this BBN-developed user interface may be found in Appendix E.

5.1.2 Screen Descriptions

Screen descriptions are structured text files which specify the appearance and position of all items seen by users on the console screen. These text files are parsed by the user interface package to produce viewable screens with which users interact. The text files also define the sequence of actions to be taken when a displayed item is selected. The user interface invokes other ASAN modules and passes data to them, as directed by user interaction with the screen descriptions. The nature and consequences of any user interaction can be changed simply by changing the relevant screen description, greatly easing the burden of system development and maintenance.

The screen-oriented approach to user interaction is especially convenient for users without advanced typing or computing skills. BBN's implementation of a screen-oriented user interface also makes provision for context-sensitive help for all displayed items. The interface further simplifies interactive use by permitting consistent limit checking and default procedures for user-entered parameter values. All displayed items modified by the user are subjected to immediate limit checks on values accepted from the keyboard, so that the user cannot provide unreasonable information.

Current values of all displayed parameters (those which will remain in force unless altered by users) are also visible at all times, and do not revert to pre-defined default values unless intentionally changed by users. This policy encourages iterative use of the software to explore alternative solutions by making it convenient to perform sets of related calculations that differ in only one or two parameters.

The user interface screens planned for the prototype version of ASAN are listed in Appendix E.

5.2 Control and Data Interface Software Details

5.2.1 User Interface

5.2.1.1 Control

At system startup, after certain housekeeping tasks (including configuration validation) are performed, the mainline routine initializes the user interface and transfers control to it. Thenceforth, the user interface remains in control and issues subroutine calls to perform all system operations. Therefore, all other system modules are written as subroutines.

5.2.1.2 Data

Data provided as "arguments" to the system modules reside in data structures (rather than being passed directly in the calls) except in rare cases; this technique results in a speed increase (since the stack need not be manipulated to store and recall arguments), allows more positive control over the data for audit and debugging, and assures that data integrity is preserved in the event of power failure.

In most systems, the user interface maintains a separate copy of data to be displayed and manipulated by the user. In ASAN, all data known to the user interface is referenced by "pointing" to common data storage outside the interface; this means that only one copy of each data item is necessary, avoiding possible inconsistency problems.

5.2.2 Data Managers

5.2.2.1 Control

The bulk of the geodata and text manager subsystem software is included without modification as supplied by the manufacturers. The supplied control software is not used, however; all interaction is handled by the ASAN user interface, by means of direct calls or small control subroutines written during ASAN development. Text data entry and viewing, using the console display screen and keyboard and pointing device, are performed by the ASAN user interface; graphic data entry and viewing use the separate graphics display monitor and pointing device, and are mediated by the geodata manager.

5.2.2.2 Data

All requests to read, modify, and/or write databased data are handled by the Geodata Access Manager and Text Data Access Manager modules. These modules maintain audit information and resolve data dependencies.

5.2.3 Analytic Modules

5.2.3.1 Control

Like all ASAN system modules, the analytic modules are run under the control of the user interface and do not communicate directly with the user. Some modules, however, have associated user interface screens that are displayed during module execution, to monitor progress and report results. Modules do not call each other directly; each module is a discrete executable entity, with control passing through the user interface between module invocations.

The module criteria database lists the input data requirements and other conditions that must be

satisfied before each module can be executed; these criteria are checked before the module is invoked. If an error occurs during module execution, an error code is set before the module returns control to the user interface.

5.2.3.2 Data

With few exceptions, all input and output data associated with analytic modules is stored in databases. Arguments are not passed to the modules in their calls, and no independent global data storage exists.

5.2.4 Checklists

Each call made between the user interface and another system module passes through the Checklist Sequencer. This module references a task-specific local copy of the appropriate checklist to keep track of which task steps have been performed, when, and in what order; anomalies are reported to the user through a user interface screen for resolution.

5.2.5 Audit

Each call made between the user interface and another system module, and each database access request, passes through the Audit Trail Maker, which records the event in the audit trail database. The recorded information includes a time stamp, a task identifier, a user identifier, a description of the event itself, and the state of the module or data element before the request.

5.2.6 Peripherals

Several hardware peripheral devices are part of the ASAN system package: a high resolution graphics controller card and graphics display monitor, a pointing device (touch sensitive screen) associated with the graphics subsystem, a pointing device (mouse) associated with the console display monitor, a printer, a tape backup controller card and tape drive, and others. These devices are controlled by software drivers which are invisible to the user, who interacts with function-oriented screens (e.g., "backup", "hardcopy", "display") rather than with the devices themselves.

5.3 Overall Control Flow

All system activity is carried out by user interaction with context-dependent menu screens, under the control of task checklists.

When a task is begun, it is given a name by the user and the system establishes a unique storage environment for it. The user delineates the area of concern, through interaction with the appropriate geodata and text databases, until the task context is satisfactorily defined.

Next, the user describes a particular problem, including information drawn from databases (mission requirements, route geometry, etc.), selected points or areas of particular interest, noise sources, and affected points or areas. Additional calculational parameters can be specified explicitly, or defaults can be assumed. The problem is given a name by the user, and the system establishes a unique storage environment for it.

The analysis phase is entered on user command. The appropriate Noise Exposure Estimation Modules are exercised to generate exposure estimates. As directed by the user or defaulted, one or more of the Effects Prediction Modules are then exercised to generate effects information. After these analyses are prepared the user can invoke the Effects Comparison Module to compare the noise effects associated with the current problem set specifications with those of other problem sets.

The report generation phase is entered on user command. Using the output from the Analytic modules, appropriate prose is generated. If desired, a complete illustrated document, suitable for publication, can be produced.

6. System Utility Modules

6.1 Checklist Manager

This module is called by other ASAN modules to keep track of system activities by reference to appropriate checklists. It gives the calling module information about which tasks in a given checklist have been completed, which tasks remain undone, which are mandatory, which are optional, which should be performed next, and so on. Checklists are available to supervise the overall environmental assessment process, to determine whether all the steps in a task have been completed before going on to the next task, to make sure all the necessary constituents of a document have been developed, to specify the actual composition of documents, and the like.

When an environmental impact assessment task is begun, the Checklist Manager is invoked and told which checklist to follow. The process calls it again after every action to update the task progress record and obtain information about the next step. The Checklist Manager saves its current state (name of task, name of checklist, position in checklist, etc.) in a group of global variables and as pointers in the stored checklist, so that the current task state is available in the event of an interruption of a computing session. Such interruptions can occur for reasons ranging from personal convenience to system malfunction or administrative query.

6.2 Audit Trail Management

Audit trail management is implemented as two software modules sharing a common data access mechanism. The first module records the details of every user-directed operation in an audit trail database. The second module uses the database to restore ASAN data structures and processing activities to a prior state.

The following information is recorded in the audit trail database when a user logs into ASAN:

```
Name of user
Date and time of day
System revision level.
```

The following information is recorded in the audit trail database when any subsequent operation is performed:

```
Date and time of day
Name of operating module
```

Description of operation.

An "undo" directive is available on all control screen displays. Activating this function causes the ASAN data structures and processing conditions to be reset to its immediately previous state. This operation is a toggle: an undo while in State One resets the system to State Two, and a succeeding undo reverts to State One.

It is also possible to restore a more distantly preceding state by rewinding through the audit trail, reversing each operation until the desired state is reached.

Practical constraints of processing power and database rollback capability will limit the capabilities available in the demonstration system. These considerations may also force different forms of implementation of this capability in later versions of ASAN from that described above.

6.3 Currency Validation Module

It is anticipated that the ASAN data structures and program modules will undergo constant revision and improvement, both during prototype development and while in operational use. It is important that the potential for introducing serious deficiencies (bugs) because of this dynamic situation be recognized and addressed. The Component Revision Level database and the Currency Validation Module are intended to mitigate potential problems of this sort.

Each of the ASAN databases, data structures, and software modules has an associated revision level number. The Component Revision Levels database lists all these system components and shows which are compatible. When ASAN is initialized, the Currency Validation Module automatically attempts to verify that the revision levels of the components actually in use conform to the revision level relationships contained in the database. If discrepancies are found, they are reported to the user as module incompatibility problems. Further system activities, other than maintenance operations to correct the incompatibilities, would be prohibited until the discrepancy is resolved..

6.4 System Backup/Restore Module

This module accomplishes backup of the information on the mass storage subsystem onto the streaming tape subsystem, and its subsequent retrieval when necessary. The module is to be largely composed of software supplied by the selected streamer tape hardware vendor.

7. Descriptions of ASAN Modules

This chapter describes ASAN's major software modules. The descriptions of the various modules vary in depth of detail according to the importance of the module for the prototype version of ASAN and the likelihood that detailed information will be available in the databases upon which certain of the calculations will depend. Descriptions provided for the algorithms are preliminary ones subject to modification during implementation of the prototype system. For example, the names of the databases may be expected to change as the entire database structure evolves.

The modules may be divided into the following four categories:

- Noise Exposure Calculation Modules.

  These modules estimate the noise exposure produced by stationary, subsonic, and supersonic airborne and near-ground sources at a point on the ground.

- Noise Effects Calculation Modules.

  These modules estimate various effects of aircraft noise exposure on people, animals, and structures.

- Report Generation Modules.

  These modules use the assessment problem specifications and the resultant calculational findings to generate prose and illustrations for incorporation into an Environmental Impact Statement (EIS), Environmental Assessment (EA) or Finding of No Significant Impact (FONSI).

- Control Logic Modules.

  These modules provide for the flow of information among the user interface, the text, numeric, and geodatabase systems, and individual modules.

7.1 Noise Exposure Calculation Modules

7.1.1 Cumulative Integrated Noise Exposure Calculation Module for Subsonic Aircraft Flyovers

FUNCTION: This module estimates the A-weighted integrated noise exposure generated by a specified set of subsonic aircraft operations at a user-selected point on the ground.

OUTPUT: This module generates floating point values of L_{dn}, 24 hour L_{eq}, and OSHA-rule (5dB per doubling of duration) cumulative exposure.

DATA REQUIRED:	SOURCE OF DATA:
Number of operations by time of day	Mission Requirements database

by aircraft type by course and speed	
SEL, level vs. distance curves	Aircraft Noise Emissions database
latitude/longitude of point of interest	GRASS

ALGORITHM: The normalized (straight-line, level flyover at standard altitude and reference day conditions), tabulated (from 100 to 20000 ft slant distance) single event level (SEL, EPNL, etc.) is available from the Aircraft Noise Emissions Database. The Mission Requirements Database supplies the deviation from the standard reference conditions as well as the number of operations.

The program calculates the integral of the single event noise exposure over time, using a first order (1/r**3) propagation model to estimate the contribution from an arbitrary flight track. This integral represents the exposure due to one daytime operation normalized to one daytime reference operation. Once calculated, the integral is adjusted for the number of operations during day, (evening), and night time as demanded by the measure being evaluated.

The integral is piecewise continuous: it is computed as the sum over the straight lines and circle arcs that make up the actual flight trajectory. The model assumes that changes in power setting vary at most linearly over a segment. The variations in power are mapped into corresponding changes in acoustic output and are included in the numerical integration. To introduce the least estimation error, the single event level table is entered at the point of closest approach.

The definition of the OSHA-rule for aircraft flyover models has not been established as of the writing of this report. For the demonstration system, implementation will be limited to:

1. Calculation of simple MTR flight operations (i.e. level operations, approximated by line segments).
2. Ability to import levels calculated outside of the system (e.g., NOISEMAP).
3. Ability to merge computed and stored data into a single map layer for display.

7.1.2 Subsonic Aircraft Single Event, Air-to-Ground Noise Propagation Calculation Module

FUNCTION: This module predicts the propagation through the atmosphere of a linear acoustic signal from an airborne source to a point on the ground.

OUTPUT: This module generates estimates of outdoor one-third octave band sound pressure level time histories from 40 Hz to 5000 Hz at half second intervals for the duration of a noise event.

DATA REQUIRED:	SOURCE OF DATA:
aircraft type	Mission Requirements database
aircraft location (lat/long)	Route segment database

aircraft altitude	Route segment database
course and speed of aircraft	Route segment database
temperature, humidity	To be determined
latitude/longitude of point of interest	GRASS

ALGORITHM:

1. Implementation of the SAE standard algorithm to propagate one third-octave band data through a standard atmosphere.
2. Implementation of USAF standard flyover data reduction programs: ignoring wind and ground impedance effects, find the 10 dB down points of haystack pattern and calculate attenuation due to spherical spreading and atmospheric absorption (by SAE method), subtract attenuation from source spectrum and integrate over time.

   The source spectrum is external data and is either inferred from a typical A/C spectrum or is user-provided (engineering estimates or measured flyover data).

7.1.3 Cumulative Integrated Noise Exposure Calculation Module for Supersonic Aircraft Flyovers

FUNCTION: This module estimates the C-weighted integrated noise exposure generated by a specified set of subsonic aircraft operations at a user-selected point on the ground, as well as peak overpressures for individual sonic booms in units of psf.

OUTPUT: This module generates floating point values of L_{cdn} and 24 hour L_{ceq}.

DATA REQUIRED:	SOURCE OF DATA:
Number of operations by time of day by aircraft type by course and speed	Mission Requirements database
latitude/longitude of point of interest	GRASS
mach # and related parameters	To be determined

ALGORITHM: Algorithm will be based on research performed and code written under NSBIT Task 1. This information is not yet available at the time of this writing. It is being developed for mainframe computation in batch mode. At some point in the future this module will be adapted for execution on microcomputers.

7.1.4 Supersonic Aircraft Single Event Noise Propagation Calculation Module

FUNCTION: This module predicts the propagation through the atmosphere of a non-linear acoustic signal (shockwave) from an airborne source to a point on the ground.

OUTPUT: This module generates predictions of one-third octave band sound pressure levels at a point on the ground.

DATA REQUIRED:	SOURCE OF DATA:
Aircraft type	user-supplied at run time
latitude/longitude of point of interest	GRASS
mach # and other parameters	To be determined

ALGORITHM: To be determined

7.1.5 Cumulative Integrated Noise Exposure Calculation Module for Near Ground Noise Sources (Aircraft and Other)

FUNCTION: This module estimates cumulative (weighted) integrated noise exposure propagated from a specified noise source (stationary or subsonic) on or near the ground to another point on the ground.

OUTPUT: This module generates floating point values of L_{dn}, 24 hour L_{eq}, and other measures appropriate for these sources.

DATA REQUIRED:	SOURCE OF DATA:
extent of source (point or line)	Noise Source database
sound exposure level of source	Noise Source database
duration of operation of source	Noise Source database
daily hours of operation of source	Noise Source database
if moving, course and speed (?)	user-supplied at run time

ALGORITHM: It is expected that the following or similar, generally used programs will be accommodated:

1. For aircraft noise sources, unobstructed field conditions implement USAF standard programs.
2. For aircraft noise sources, obstructed field conditions implement a modification of (1) or an appropriate new model. No such models are contemplated as part of ASAN development.
3. For highway sources possibly implement ADRPM (or similar) program or provide ability to import data generated thereby.
4. For other noise sources under unobstructed conditions, SAE model for

atmospheric propagation is used. For obstructed field conditions modules will have to be prepared at some point in the future as warranted.

7.1.6 Single Event Noise Calculation Module for Near Ground Noise Sources (Aircraft and Other)

FUNCTION: This module predicts the propagation through the atmosphere of an acoustic signal from a stationary or subsonic noise source on or near the ground to a point on the ground.

OUTPUT: This module generates estimates of outdoor one-third octave band sound pressure level time histories from 40 Hz to 5000 Hz at half second intervals for the duration of a noise event.

DATA REQUIRED:	SOURCE OF DATA:
extent of source (point or line)	Mission Requirements database
sound exposure level of source	Noise Source database
duration of operation of source	Mission Requirements database
daily hours of operation of source	Mission Requirements database

ALGORITHM: No generally applicable models exist at this time. Individual models will have to be developed as warranted.

7.1.7 Ambient Noise Environment Estimation Module

FUNCTION: This module estimates one-third octave band means and variances of outdoor ambient noise levels at a specified point over the frequency range of 40 - 5000 Hz.

OUTPUT: This module generates floating point values of triplets of means, variances, and standard errors of the mean for one-third octave band sound pressure levels of the outdoor ambient noise distribution from 40 Hz to 5000 Hz. The module also produces a set of references to a set of numbered notes describing the assumptions needed to generate its estimates, as follows:

DATA REQUIRED:	SOURCE OF DATA:
position of point	GRASS
empirical measurements	Noise Measurement database
population density of area within a radius of 1 km of point	Census Tract database
locations of all noise sources within 1 km of point	Noise Source database
emission levels of all noise sources within 1 km of point	Noise Source database

local land-use | Land Use database

ALGORITHM: This module develops an estimated ambient noise spectrum from several disparate types of information:

1. Ambient noise levels estimated from field measurements contained in a Noise Measurements Database.
2. Ambient noise levels inferred from population density estimates and general assumptions about the usual spectral shape of outdoor ambient noise distributions derived from Fidell, Horonjeff, and Green (1981).
3. Ambient noise level estimates based in part on the contribution of noise from specifically identified non-aircraft sources.

First, this module checks to see if any field survey values are available in the Noise Source Database. If any are available, they should be adopted in preference to any estimates. Some processing of empirical measurements may be necessary to convert them into third octave band mean levels, and to estimate default values for variances and standard errors if unavailable.

Absent empirical measurements, the module should exercise the 10 log ρ + 22 dB rule of thumb (Galloway, Eldred, and Simpson, 1973) to estimate L_{dn} in the area within about +/- 5 dB). Then it must assume a spectral shape (probably falling at 6 dB/octave from a peak in the vicinity of 160 Hz). The assumed spectral shape should probably be checked for consistency against the Land Use Database to make sure it is reasonable. Finally, the contributions of any major noise sources (heavily trafficked highways, industrial sources, etc.) should be estimated and added logarithmically to the estimates. A description of the method used to arrive at the estimate, complete with any assumptions adopted, should be produced as well.

7.2 Noise Effects Calculation Modules

The external appearance of all the noise effects calculation modules is consistent from the point of view of the control layer. This allows for reduced cost and complexity of software construction and maintenance, and enables modules to be added, replaced or improved with ease. The important aspects of uniformity include the following:

- Input passed as calling arguments consists of a "depth of analysis" value, a pointer to the relevant exposure estimate(s), and a pointer to the location where output is to be stored.

  All noise effects calculation modules operate at several depths of analysis. At the most superficial depth of detail, they produce "quicklook" estimates based on simplifying assumptions and default values for most input variables. Individual modules differ in the depth of alternative analyses available, depending both on the availability of meaningful exposure information and on the availability of sufficient theoretical understanding on which to base calculations.

- All other input is obtained from system-wide global variables and default parameters, or derived from the data structures created during the Problem Definition phase.

- Output returned to the calling program includes a "success code," a "severity of effect" code, and a pointer to any module-specific output.

  The "success code" indicates the outcome of the calculation software execution process and defines contents of the resultant module-specific data block; it has values of the following form:

```
0 - execution is not possible because the available input
    is incomplete or improper.
1 - the precision of the estimate is satisfactory.
2 - the precision of the estimate is unsatisfactory.
```

  The "severity of effect" code provides a summary judgment of the calculational results; it has values of the following form:

```
0 - effect not considered in the current analysis
1 - magnitude of predicted effect is inconsequential
2 - magnitude of predicted effect is of minor importance
3 - magnitude of predicted effect is of considerable importance
4 - magnitude of predicted effect is of great importance
```

- The actual calculational output is returned as a group of data blocks. Each such data block contains a keyword representing a text string, and a list of metrics with associated uncertainty values.

7.2.1 Habitability Interpretation Module

FUNCTION: This module compares an L_{dn} value with criteria adopted by U.S. federal agencies for land use compatibility.

OUTPUT: This module generates a numeric code representing one of the following sorts of text strings:

"acceptable"
"normally acceptable"
"normally unacceptable"
"unacceptable"

for insertion by a report production module into a carrier phrase of the following sort:

"Noise exposure in the area called _(A)__ is considered _(B)__ for _(C)___"

where (A) is a phrase of the form "geographic placename," (B) is a phrase of the form "normally acceptable," and (C) is a phrase of the form "residential/industrial/recreational land use."

DATA REQUIRED:	SOURCE OF DATA:
L_{dn} estimate for an area	descriptive statistic based on grid of L_{dn} point estimates in temporary data file
Geographic place name	Political Boundary database
Habitability criteria	Habitability Criterion database

ALGORITHM: This module will perform simple table look-ups to compare estimated with tabled

L_{dn} values for each criterion, and generate one text string for each land use category of each criterion.

7.2.2 Sensitive Land Use Identification Module

FUNCTION: This module rank orders all land uses within a specified area by sensitivity to noise exposure.

OUTPUT: A list of place names within the specified area categorized and rank ordered by sensitivity to land use (most sensitive first, least sensitive last).

DATA REQUIRED:	SOURCE OF DATA:
definition of area	user-specified area
list of place names	Political Boundary database
list of land uses	Land Use database
sensitivity of land uses	Habitability Criterion database

ALGORITHM: This module determines for each place name in the Political Boundary Database whether any portion of each entry falls within the latitude/longitude coordinates of the user specified area. It then prepares a list of all land uses within the portions of any places that lie within the specified area. It finally rank orders the land uses within places by table look up in the Habitability Criterion Database. Land uses of equal sensitivity within qualifying areas are aggregated within categories in alphabetical order, and categories are assigned ranks.

7.2.3 Speech Interference Calculation Module

FUNCTION: This module calculates several indices of speech interference produced by noise exposure in a specified area.

OUTPUT: A fully developed version of this module should be able to produce a variety of indices of speech interference, including the following:

1. reduction in distance for relaxed conversation inside a residence.
2. percentage of time increased vocal effort is required.
3. estimated number of words interfered with in a 24 hour period.
4. percentage of time communication must pause.
5. average articulation index value.

DATA REQUIRED:	SOURCE OF DATA:
One third octave flyover spectra at half second interv	Noise source database
Estimated background noise	Noise source database
Numbers of overflights	Mission requirements database

ALGORITHM: The basic calculation to be performed by this module is that described in the ANSI Articulation Index Standard (S3.5-1969).

7.2.4 Prevalence of Annoyance Calculation Module

FUNCTION: This module predicts the proportion of a community highly annoyed by long term integrated noise exposure in a specified area.

OUTPUT: This module generates a nominal estimate and fiduciary limits of the proportion of a community highly annoyed, for insertion into a carrier phrase of the following form:

> "Noise exposure in the area called _(A)__ may be expected to annoy approximately __(B)__ % of the residents to a consequential degree. The approximate error of this estimate is __(C)__%"

where (A) is a phrase of the form "geographic placename," (B) is an integer between 5 and 99, (C) is an integer between 1 and 100

DATA REQUIRED:	SOURCE OF DATA:
L_{dn} estimate for an area	descriptive statistic based on grid of L_{dn} point estimates in temporary data file
Geographic place name	Political Boundary database
population density	Land Use database

ALGORITHM: The module evaluates a third order polynomial derived by Schultz (1978) to derive the nominal estimate, and an error function to establish error bounds.

7.2.5 Sleep Interference Calculation Module

FUNCTION: This module predicts the expected number of awakenings associated with the integrated detectability of nocturnal noise intrusions.

OUTPUT: This module produces a range estimate for the expected number of awakenings per night in a specified area, for insertion into a carrier phrase of the following form:

> "Noise exposure produced by __(A)__ in the area called _(B)__ may be expected to produce between __(C)__ and __(D)__resident- awakenings per night."

where (A) is a name of a noise source, (B) is a phrase of the form "geographic placename," (C) is an integer lower bound of the nominal estimate rounded to the next lowest order of magnitude, and (D) is an integer upper bound of the nominal estimate rounded to the next highest order of magnitude.

DATA REQUIRED:	SOURCE OF DATA:
name of noise source of interest	Noise Source database
median value of distribution of outdoor one-third octave band noise levels from 40 to 5000 Hz of ambient noise distribution	Outdoor Ambient Noise database

10th centile (L_{90}) value of distribution of outdoor one-third octave band noise levels from 40 to 5000 Hz due to noise source of interest	To be determined
nightly number of noise intrusions produced by sources of interest	Mission Requirements database
average duration of noise intrusions produced by sources of interest	To be determined
total population of specified area	Census Tract database

ALGORITHM: A one-third octave band insertion loss spectrum for a residence is subtracted from quantities A and B to estimate their indoor sound pressure levels. The resulting one-third octave band sound pressure levels for quantity A are added on an energy basis (logarithmic sum) to the assumed median values of a distribution of indoor ambient noise levels.

A combined detector efficiency and critical bandwidth adjustment is then applied to the 50th centile indoor ambient and 90th centile indoor noise source one third octave band levels to estimate the detectability of noise intrusions in the bedroom. The integrated detectability (in units of d'-seconds) for a single noise intrusion is then calculated by multiplying the estimated detectability by the average duration of a noise intrusion.

The probability of awakening associated with the integrated detectability of a single noise intrusion, derived from a relationship published by Horonjeff, Fidell, Teffeteller and Green (1982), is then multiplied by the number of such noise intrusions to estimate an expectation for the number of nightly awakenings. This number is rounded down to the next lowest order of magnitude and up to the next highest order of magnitude to produce an expected range of numbers of awakenings per night for a single resident. These upper and lower bounds are then multiplied by the population of the specified area to produce the reported estimates.

7.2.6 Hearing Damage Risk Assessment Module

FUNCTION: This module assesses hearing damage risk associated with noise exposure in a specified area.

OUTPUT: This module produces a nominal estimate for the likelihood of hearing damage for insertion into a carrier phrase of the form:

> "Noise exposure produced by __(A)__ in __(B)__poses __(C)__risk of hearing damage to the resident population."

where (A) is the name of a noise source, (B) is a geographic place name, and (C) is a one of the following strings:

> "no meaningful"
> "a possible"
> "a considerable"

DATA REQUIRED:	SOURCE OF DATA:

A-weighted equivalent level	integrated exposure calculation module
Impulse noise spectra	Supersonic aircraft single event noise propagation calculation module
name of noise source	Noise source database
duration of exposure	Mission requirements database

ALGORITHM: This module compares estimated exposure with two published criteria for hearing damage risk: OSHA's criterion for continuous noise, and CHABA's criterion for impulsive noise. Since OSHA's hearing damage risk criterion assume forty years of exposure, 250 days per year, for 8 hours per day, a correction factor must be calculated to adjust the OSHA criterion to the expected duration of exposure to noise of the proposed action. The CHABA criterion for hearing damage risk due to impulsive noise exposure is written in terms of minutes of daily exposure over a ten year period; these figures must also be adjusted in a similar manner. The contents of the text string "C" in the above carrier phrase will be set to "no meaningful" for estimated exposure values that do not come within 10 dB of either the impulsive or continuous hearing damage risk criteria; to "a possible" for estimated exposure values within 10 dB of the criteria; and to "a considerable" for values that meet or exceed the criteria.

7.2.7 Glass Breakage Model

FUNCTION: This module predicts the approximate number of claims that may be expected for glass breakage due to sonic boom exposure.

OUTPUT: A nominal value for the estimated number of claims.

DATA REQUIRED:	SOURCE OF DATA:
approximate number of structures in area	land use database
expected number of supersonic operations	mission requirements database
distribution of overpressures	supersonic aircraft single event noise propagation module

ALGORITHM: An estimate of number of expected claims will be based on evaluation of a polynomial relationship derived from a model of window breakage as functions of number and level of sonic booms.

7.2.8 Endangered Species Reproductive Success Estimation Module

FUNCTION: This module reports any evidence that noise exposure in an area interferes with reproductive success or population size of an endangered species.

OUTPUT: A text string.

DATA REQUIRED:	SOURCE OF DATA:
list of endangered species living within a specified area	GRASS map layer
published reports of interference with reproductive success	citation index database

ALGORITHM:

7.2.9 Livestock Economic Damage Estimation Module

FUNCTION: This module reports any evidence that noise exposure in an area might generate claims for economic damages involving livestock.

OUTPUT: A text string.

DATA REQUIRED:	SOURCE OF DATA:
list of prior claims for economic damages to livestock within a specified area	GRASS map layer, legislative database
published reports of susceptibility of economic loss to livestock due to aircraft noise exposure	citation index database

ALGORITHM: This module will identify all areas in a land use database in which the potential exists for high level noise exposure of livestock, and report any such areas in which prior economic damage claims involving livestock have been paid by the USAF.

7.2.10 Effects Comparison Module

FUNCTION: This module compares the effects produced by each of the other effects calculation modules in accordance with a set of rules establishing a hierarchy of significance of effects to produce a rank ordering of a set of predicted effects.

OUTPUT: This module produces a text string of the following form:

"__(A)__ of the predicted effects of noise exposure produced by __(B)__ in __(C)__are potentially important. In order of significance, they are __(D)__ through __(E)__. The following effects were not considered for lack of information in this analysis: __(F)__ through __(G)__."

where (A) is a positive integer expressed as a character string (i.e., "None," "One," "Two," etc.; (B) is a noise source name, (C) is a geographic place name, and (D) through (G) are names of noise effects.

DATA REQUIRED:	SOURCE OF DATA:
Noise exposure effects estimates	Noise exposure effects calculation modules

ALGORITHM: A set of heuristics will be applied to the outputs of the various noise exposure effects calculation modules to prioritize them. Any risk of hearing damage will be accorded the highest priority. The remainder of the ranking of the relative importance of predicted effects of different relative magnitudes will be implemented as a simple set of if-then rules.

7.3 Report Generation Modules

7.3.1 Effects Prose Generator

FUNCTION: Uses information generated by effects calculation modules to access the Boilerplate Database and produce prose descriptions of the effects.

OUTPUT: Prose fragments.

DATA REQUIRED:	SOURCE OF DATA:
Module output classification keys	Module Criteria database
Module output values	Calculation Results database
Module output success code	Calculation Results database
Effect Severity Code	Calculation Results database

ALGORITHM: The information generated by the effects calculation modules is in the form of a group of global data blocks, each containing a keyword text string and a list of numerical or textual metrics. The Boilerplate Database is accessible by index on effect and severity so that records can be related to effects assessment. These records include fragments of prose text containing "placeholders" (data format descriptors). For each analysis output data block, the Effects Prose Generator searches the Boilerplate Database until a matching keyword is found. The placeholders in the associated prose fragment are replaced by the metrics from the data block, and the completed prose fragment is stored.

7.3.2 Prose Optimizer Module

FUNCTION: Reorganizes and rearranges text fragments into less redundant and more understandable prose.

OUTPUT:

DATA REQUIRED:	SOURCE OF DATA:
Effects prose fragments	Calculation Results DB

ALGORITHM:

NOTE 1: This module will not be included in the demonstration prototype system, but will be developed later in the project.

7.3.3 Word Processor Module

FUNCTION: Allows the user to rearrange and edit the text materials in databases or generated reports.

OUTPUT:

DATA REQUIRED:	SOURCE OF DATA:
Document prose	Task Document DB

ALGORITHM: An already developed word processing facility will be integrated into ASAN. The selected word processor will be:

- screen oriented (so that "What You See Is What You Get").
- organized so that most common operations can be carried out with a small set of easily learned commands.
- able to perform common operations rapidly and efficiently.
- able to perform "multiple buffer" and "multiple simultaneous file" operations, in which text from several separate sources can be manipulated at once.
- able to operate according to predefined programs or "macros" written in an understandable dialect.
- callable as a library of functions from external C language programs.
- able to operate while physically co-resident in main memory with other software, and take advantage of whatever memory is available.
- an efficient user of system resources, especially mass storage and the console display screen.
- capable of handling very large files.

7.3.4 Document Formatting Module

FUNCTION: Formats and beautifies text produced by ASAN for hardcopy output.

OUTPUT:

DATA REQUIRED:	SOURCE OF DATA:
Document prose	Task Document DB

ALGORITHM: An already developed, device independent document formatting facility will be integrated into the system. This facility will:

- operate by decoding a rich set of formatting directives, in fairly simple natural language, embedded in the document text.
- handle documents containing mixed text and high resolution color graphics.
- be device independent so that output can be generated for many types of screen display and hard copy devices (including those using the PostScript publication protocol standard) or as plain text files containing no unusual control characters.
- be able to perform "join" operations, in which input from several separate sources can be combined to produce a single document.
- be able to operate according to predefined document templates written in a simple dialect.
- be implemented as a library of functions callable from external C language programs.
- be able to operate while physically co-resident in main memory with other software, taking advantage of whatever memory is available.
- be able to use system resources efficiently, especially mass storage.
- impose no serious limitations on the maximum size of a database.
- implement a comprehensive set of default parameters.
- not require user commands for common operations.

7.4 Control Logic Modules

7.4.1 Main Program

FUNCTION: Initial Entry to ASAN

OUTPUT: None

DATA REQUIRED:

None

ALGORITHM:

1. Calls Currency Validation Module,
2. Calls Startup Module,
3. Transfers control to user interface.

7.4.2 Currency Validation Module

FUNCTION: Check release level of implementation against Configuration Database.

OUTPUT: Sets proceed flag or puts up error info screen.

DATA REQUIRED:	SOURCE OF DATA:
Module and database revision levels	Revision Level DB

ALGORITHM:

7.4.3 Startup

FUNCTION: Configure system environment as specified by command file.

OUTPUT: This module identifies the peripherals and other resources to the ASAN software, and records these data in a System Resources Database.

DATA REQUIRED:	SOURCE OF DATA:
System resource identifications	Startup file

ALGORITHM:

7.4.4 User Interface

FUNCTION: This module mediates all user interaction with the ASAN system through the console display monitor, keyboard, and pointing device (mouse).

OUTPUT:

1. User Interactive Console Screens.
2. Call Lists

ALGORITHM: See the detailed discussions of the user interface elsewhere in this Report.

7.4.5 Geodata Access Manager

FUNCTION: Interface between applications and geodata manager (GRASS) which controls and maintains audit and data integrity.

OUTPUT:

1. Geodata Manager calls to read, modify, and write geodata records.
2. Updates to data dependency database.

DATA REQUIRED:	SOURCE OF DATA:
Data dependencies	Data Dictionary
	Data Dependencies DB

ALGORITHM:

To READ a data element, NOT for use in a calculation:

1. Return the current value.

To READ a data element, for use in a calculation:

1. Confirm that the current value has not been invalidated by more recent updates to other data elements upon which this value depends.
2. Return the current value.

To ADD a new data element:

1. Create an Audit record, containing NULL as the data value before modification.
2. Write the new value into the database.

To MODIFY (update or delete) an existing data element:

1. Create an Audit record, containing the data value before modification.
2. Write the new value into the database.

7.4.6 Text Data Access Manager

FUNCTION: Interface between applications and text data manager which controls and maintains audit and data integrity.

OUTPUT:

1. Text Data Manager calls to read, modify, and write text data records.
2. Updates to data revision level tables

DATA REQUIRED:	SOURCE OF DATA:
	Data Dictionary
	Data Dependencies database

ALGORITHM:

To READ a data element, NOT for use in a calculation:

1. Return the current value.

To READ a data element, for use in a calculation:

1. Confirm that the current value has not been invalidated by more recent updates to other data elements upon which this value depends.
2. Return the current value.

To ADD a new data element:

1. Create an Audit record, containing NULL as the data value before modification.
2. Write the new value into the database.

To MODIFY (update or delete) an existing data element:

1. Create an Audit record, containing the data value before modification.
2. Write the new value into the database.

7.4.7 Task Initialization Module

FUNCTION: Initialize task-specific databases and checklists for new assessment scenarios, or load an existing scenario.

OUTPUT:

DATA REQUIRED:	SOURCE OF DATA:
	Task-specific Context DB

ALGORITHM: Disk storage is allocated, and the data structures necessary to perform the selected assessment task are copied into the allocated space. The current system operating context is configured to map to the newly created task space.

7.4.8 Checklist Manager

FUNCTION:

1. Update a task's checklist in progress to reflect current status and subsequent steps.
2. Determine whether all analyses for a task have been completed.
3. Display status of present assessment scenario.

OUTPUT:

1. User screens.
2. Updates to the task's status flags.

ALGORITHM:

7.4.9 Module Criteria Checker

FUNCTION: Determine whether the data and conditions necessary to permit the invocation of an analytic module have been met.

OUTPUT:

1. User screens (only if anomalies are encountered).
2. Updates to the task's status flags.

DATA REQUIRED:	SOURCE OF DATA:
Module name	User Interface
Current checklist status	Checklist database
Module criteria	Module Criteria database
Data dependencies	Data Dependency database

ALGORITHM: The criteria are looked up and checked against the current checklist status. If the necessary conditions are met, the module's required data is validated for currency by reference to the Data Dependency Database. If no anomalies are encountered, the module is allowed to execute.

7.4.10 System Maintenance ("Housekeeping")

FUNCTION:

1. Archiving of Information.
2. Reloading Archived Information.
3. Recovery from Machine Failure.
4. Installation of New Global Databases.
5. Local Corrections and/or Updates to Global Data.

OUTPUT:

ALGORITHM:

7.4.11 Audit Trail Maker

FUNCTION: Create and manage (limit length of) Audit Trail Database.

OUTPUT: Audit Trail Database

DATA REQUIRED:	SOURCE OF DATA:
arguments passed by calling procedures, including: name of database modified old sequence number of database record number in database field number within record old value of field	every other module

ALGORITHM: Write and timestamp a record to Audit Trail Database. If the space available for audit trail storage is exceeded: allow the user to discard or backup early portions of the audit trail.

7.4.12 Audit Trail Rewinder

FUNCTION:

1. rewind the audit trail
2. failure recovery
3. prepare summary reports

OUTPUT: State of the databases at step n.

DATA REQUIRED:	SOURCE OF DATA:
number of steps to unwind	user input
audit trail	Audit Trail database

ALGORITHM: By reading the audit trail "backwards" in time from the most recently recorded position, "undo" each operation until the specified point has been reached.

7.4.13 Audit Trail Display

FUNCTION: Display audit trail.

OUTPUT: Formatted screen display of audit trail, evocation of audit trail rewinder.

DATA REQUIRED:	SOURCE OF DATA:
audit trail	Audit Trail database
portion of trail to display	user input

ALGORITHM: Generate scrollable screen display of formatted audit trail data with ability to select a particular audit trail record via cursor keys.

7.4.14 Global System Currency Validation Module

FUNCTION: Check version numbers of modules against Revision Level Database for consistency; inhibit system function as necessary.

OUTPUT: message to user, sets ok to proceed flag

DATA REQUIRED:	SOURCE OF DATA:
	revision level database
	executable modules

ALGORITHM: Runs once at system startup to check all modules and databases to prevent operation of system with inconsistent modules and databases.

7.4.15 Data Consistency Checker

FUNCTION: Protect work in progress against inconsistent databases.

OUTPUT:

DATA REQUIRED:	SOURCE OF DATA:
Data Dependency DB revision level	Configuration database
Data locations	Data Dictionary
Data dependencies	Data Dependency database

ALGORITHM: To be determined

7.4.16 Hardcopy Output Module

FUNCTION: Direct text and graphic output to a specific hardcopy device selected by the user at run time.

OUTPUT: Hard copies.

DATA REQUIRED: SOURCE OF DATA:

ALGORITHM: It supports a variety of graphic display and hard copy hardware, formatting and transmitting the data in a manner appropriate for each. It implements spooling to pass graphic outputs to the selected device without tying up the host processor, to permit productive use of the environmental planner's time while output is in progress.

8. Descriptions of ASAN Databases

This chapter lists some of the databases upon which ASAN will operate. A final comprehensive description of all ASAN databases and their contents will be developed later in the project.

8.1 Centrally Maintained Geodatabases

8.1.1 Maps and Charts

Aeronautical Sectional Charts
Road Maps
Land Use Maps
Defense Mapping Agency Maps
Topographic Maps
Census Tract Maps
Political Jurisdiction Maps
Archaeological Sites
Average Weather/Climate Charts
Existing Air Base/MOA/MTR Maps
Proposed MOA/MTR Maps

8.1.2 Imagery

Remote Sensing Imagery
Aerial Photographs

8.2 Locally Maintained Geodatabases

8.2.1 Maps and Charts

Utility Districts
Mail Routes
Property Tracts
Historical Sites
Local Flight Track Maps

8.2.2 Imagery

Local Area Aerial Photographs

8.2.3 Maps Produced During the Environmental Assessment Process

Proposed Routes
Composite Maps

8.3 Centrally Maintained Text Databases

8.3.1 Numerical Data

Aircraft Noise Emissions Data

8.3.2 Noise Exposure & Impact References

Citation Indices & Summaries
Boilerplate Text

8.3.3 Checklists

Overall Environmental Planning Process
Geodata Collection
Text Data Collection
MOA Specification Contents
MTR Specification Contents
Analysis Process
EA Preparation
EIS Preparation
FONSI Preparation

8.3.4 Document Templates

```
Progress Reports to Headquarters
Form Letters
EIS
FONSI
```

8.4 Locally Maintained Text Databases

```
Points of Contact
Mission Requirements
Draft Documents
Final Documents
System Problem Reports
```

8.4.1 Output from the Environmental Assessment Process

```
Previous Analysis Results
Current Analysis Results
```

8.5 Internal System Text Databases

```
Component Revision Levels
Analysis Output Data Blocks
Audit Trail
```

References

Dickinson, John (1986) "Programmable Relational Databases", PC Magazine, June 24, 114-227.

Federal Systems Division, Wang Laboratories Inc. (1986) "Resource Guide for the AMMUS Contract (Air Force Minicomputer Multi-User System)", Contract Number F19630-86-D-0001.

Fidell, S., and Harris, M. (1987) "Design Recommendations for Environmental Planning Aid Software", BBN Report Number 6342.

Fidell, S., Horonjeff, R., and Green, D.M., (1981) "Statistical Analyses of Urban Noise", Noise Control Engineer, Vol. 16, No. 2, 75-80.

Galloway, W. (1981) "Assessment of Community Response to High Energy Impulsive Sounds", Report of Working Group 84, Committee on Hearing, Bioacoustics, and Biomechanics, National Research Council, National Academy of Sciences, Washington, D.C.

Harris, M. (1986) "U -- A Universal Interactive Interface", unnumbered BBN Report.

Horonjeff, R., Fidell, S., Teffeteller, S., and Green, D.M. (1982) "Behavioral Awakening as Functions of Duration and Detectability of Noise Intrusions in the Home", J. Sound and Vib., Vol. 84, No. 2.

Parker, S., Limp, W., Farley, J., and Johnson, I. (1986) "Database Management, Geographic Information Systems, and Predictive Modeling in the Forest Service", Arkansas Archaeological Survey.

Schultz, T.J. (1978) "Synthesis of Social Surveys on Noise Annoyance", J. Acoust. Soc. Amer., 4(2), 377-403.

Appendix A
Selection of Host Computer

The portion of the prototype version of ASAN made available to end users is intended to be hosted at least in the short term on the Zenith Z-248 personal computer under the MS-DOS operating system. The reasons for this selection are:

- These machines are of reliable and proven design, and have been purchased in large quantities by the Air Force. Since they are already part of the Air Force inventory, they are likely to be available to environmental planners at the base level without special procurement efforts.
- They are sufficiently fast and powerful to support the required tasks locally, in a stand-alone environment, without extensive interaction with centralized facilities.
- They provide fully compatible upgrade paths from the Intel 80286 to the Intel 80386 processor, and from the present single-user, single task operating system to a concurrent (background/foreground) operating system which is expected to be announced in 1988. These hardware and software upgrade paths minimize the effort required to accommodate future advances in personal computer technology which the Air Force is likely to adopt by the time ASAN is ready for distribution.
- They are already the host computers for much of the existing (commercially available and previously developed) software to be included in ASAN.
- They represent a significant market force, so that many applications programs, including a number developed under Air Force sponsorship, are available in compatible versions.
- The additional peripheral cards and devices necessary to support ASAN are available relatively inexpensively to fit the machines' standardized electronic connection bus.
- Since the machines and their MS-DOS operating systems are in common use in the Air Force, management, training and documentation problems are small with respect to those that would attend selection of a different computing environment.
- The installed base of this and other IBM PC/AT compatible computers, both within the armed services and in the commercial arena is very large, assuring that their components and design philosophy will be supported well into the future.

A.1 Consideration of Alternative Host Computer System

Alternative host computer systems, described by the Federal Systems Division of Wang Laboratories, Inc., were also considered but found to be inappropriate for present purposes. These systems, provided to the Air Force under Contract F19630-86-D-0001, consist of two groups of hardware products.

1. The "VS Computers" are claimed to be equivalent in processing power to Digital

Equipment Corporation's VAX products, and are more powerful and costly than the environmental assessment task requires. These machines are intended for multi-user office automation tasks. The WANG VS configuration guide does not list most of the hardware needed for ASAN, such as high resolution graphics, touch screens, and pointing devices. Since the multiplicity of suppliers and economies of scale which operate to provide high quality, low cost peripheral devices for IBM PC/AT compatible systems are absent for AMMUS systems, the availability of these devices is likely to be poor and their cost high.

Furthermore, the VS Computers use a proprietary "virtual storage operating system." It is unlikely that the third party software to be utilized in ASAN has been written for, or can be easily translated to, this operating system; its development would add significantly to the cost and elapsed time necessary to complete the project.

There is a Unix equivalent operating system available for the VS Computers. Its use would enable the use of a wider range of third party software; however, this operating system is not covered by the AMMUS contract and hence would be more difficult and costly to acquire and maintain. In addition, the user would be required to learn an entirely new operating environment.

2. The second Wang AMMUS product group, "Programmable Workstations" (also referred to as "Wang PCs"), is built around IBM PC equivalent machines using the Intel 8086 microprocessor. Although IBM compatible third party hardware and software is explicitly supported, an 8086-based machine would not be sufficiently powerful or flexible to perform the necessary operations with acceptable speed.

Appendix B
Hardware Subsystems for ASAN prototype

The hardware subsystems described below must be added to the basic host computer to provide the full hardware capability needed for ASAN. Not all of this hardware capability is needed for the prototype version of ASAN.

B.1 Mass Storage Subsystem

The basic Z-248 host system will include a non-removable hard disk for system software storage. An additional mass storage device will be necessary to provide adequate storage space and reasonable access speeds for the databases which constitute the heart of ASAN. This additional storage will be removable to simplify updating and to allow the environmental planner to switch among several ongoing tasks. It is expected that the additional storage capability will be provided by removable magnetic disks in the demonstration system.

Digital optical disks are probably more suitable for future versions of ASAN. These devices can store approximately 500 megabytes in a small physical space, are less expensive than comparable quantities of magnetic storage, and are fairly rugged.

B.2 Graphics Display Subsystem

An add-on color graphics card and color graphics display monitor is necessary to permit display of graphic information. The card, an Imagraph IP1076-10-N- A-1.2-PC, provides resolution of 1024 by 768 pixels, displays as many as eight bit planes, and has the ability to conveniently accept input from the graphics input subsystems described below.

The graphics display device is a 19 inch (diagonal measurement) monitor distinct from the operator's console monitor that displays flicker-free full color images.

B.3 Graphics Input Subsystems

These add-on graphics input devices are needed to support ASAN's graphics manipulation capabilities.

B.3.1 Touch Sensitive Screen

A touch sensitive screen mounted directly on the graphics display monitor allows the environmental planner to indicate points or areas of interest and select displayed features rapidly and unambiguously.

B.3.2 Graphics Tablet

A graphics tablet permits entry of local maps and other data by direct tracing.

B.3.3 Console Pointing Device

The basic Z-248 personal computer must include a console display subsystem consisting of a controller card (functionally equivalent to the IBM Enhanced Graphics Adapter) and an associated display monitor (NEC Multisync or equivalent). Users will view ASAN control screens and text on this monitor.

A pointing device for the console display (Microsoft Bus Mouse or equivalent) is needed to permit efficient interaction with the screen-oriented user interface.

B.4 Digitizing Camera

Although it is not necessary for the demonstration prototype, a digitizing camera may be included in future versions of the system. This device would allow photographic images, maps and other graphic data to be entered into ASAN databases rapidly and directly. An administrative decision based on organizational and cost issues will be needed to determine whether the Air Force would prefer to centralize this capability or provide it to end users.

B.5 Hardcopy Subsystems

Capabilities for producing rough and final versions of text and graphics output products are required. The ability to produce final versions of text and graphics of sufficiently high quality for publication is likely to be too expensive to provide to every end user. Centralized reproduction facilities might therefore be needed to accommodate final publication needs.

However, every field installation of the system requires the ability to produce rough drafts and proof copies during the assessment process. A color thermal printer is recommended for this purpose.

B.6 Streaming Tape Subsystem

A streaming tape controller card and drive assembly are needed to provide efficient and reliable mass storage backup capability. This subsystem must be able to:

- operate according to straightforward and understandable control commands;
- fit all the files on the mass storage device onto a single tape cartridge;
- back up and restore all files on the mass storage device, or just those files modified since the last backup was performed, or selected groups of files, or individual files.

Furthermore, the tape backup system must be supplied with vendor developed and supported software, providing all necessary functions in the form of C language callable subroutines, and must use standard tape cartridges, preferably in a standardized recording format.

Appendix C
Effects Prose Generation

The Effects Prose Generator Module uses output produced by noise effects calculation modules to select and complete appropriate prose fragments. Text from the Boilerplate Database is merged with the generated prose fragments to produce text descriptions and documentation of the effects. This process is described in more detail below.

Each of the effects modules also categorizes its predicted effects in terms of significance with respect to a decision regarding the justification for an Environmental Impact Statement (EIS) or a Finding of No Significant Impact (FONSI). This information is stored in the form of a "significance code" as follows:

0 - effect not considered in current analysis
1 - inconsequential
2 - of minor importance
3 - of considerable importance
4 - of great importance

Prose fragments are merged with the appropriate Boilerplate Database to produce text fragments. Each Boilerplate Database record contains an effect datablock, a fragment of prose text containing "placeholders" (data format descriptors). For each datablock, the Effects Prose Generator searches the Boilerplate Database for the text appropriate to the effect and level of severity. The placeholders in the associated prose fragment are replaced by the metrics from the datablock and a completed text fragment is output.

For example, suppose the following analytic results:

MODULE: Noise Exposure Prediction Module

MODULE OUTPUT:

48_

Where "_" is an embedded directive indicating A-weighted integrated noise exposure.

TEXT PRODUCED:

"48 L_{dn}"

and

MODULE: Glass Breakage Module

MODULE OUTPUT:

10

Where 10 represents the approximate number of claims due to sonic boom exposure.

TEXT PRODUCED:

"The estimated number of claims stemming from broken or cracked glass is on the order of 10 per one hundred events."

The Boilerplate Text database also contains factual text fragments excerpted from reliable and attributable sources, which are generated to accompany the calculational results:

EFFECT KEYWORDS:
sonic boom impact; glass breakage

TEXT PRODUCED:
By far, the largest percentage of sonic boom damage claims stems from broken or cracked glass.

TEXT SO FAR: "By far, the largest percentage of sonic boom claims stems from broken or cracked glass. The estimated number of claims expected from broken or cracked glass is approximately 10 per one hundred events."

Effect datablocks can also trigger production of tables and charts to summarize noise effects. For example:

```
EFFECT KEYWORDS:    probability of glass breakage; table

TEXT PRODUCED:

-------------------------------------------------
     Overpressures          Probability of Breakage
     -------------          -----------------------
          1 psf             .000001*
          2 psf             .000023
-------------------------------------------------
       *1 pane in 1,000,000 panes
```

If the predicted effect is categorized as inconsequential the following Boilerplate text fragment might be accessed:

EFFECT KEYWORDS:
statement of no significant impact; noise

TEXT PRODUCED:
"Noise impacts for the project as proposed during operations are not significant."

or if the predicted effect is considered important the following fragment could be accessed:

EFFECT: sonic boom impact; significant

TEXT PRODUCED:
"Over a period of time a number of windows can be expected to be broken or cracked as a result of sonic booms. The Air Force has established procedures to recover the costs of damage resulting from sonic booms."

The total boilerplate text produced by the above examples (when the impact is categorized as significant) would appear as follows:

"By far, the largest percentage of sonic boom claims stems from broken or cracked glass. The estimated number of claims expected from broken or cracked glass is approximately ten per one hundred events. Over a period of time a number of windows can be expected to be broken or cracked as a result of sonic booms. The Air Force has established procedures to recover the costs of damage resulting from sonic booms."

Appendix D
Database Management Software and Principles of Database System Design

Most computer programming has, until relatively recently, been algorithm driven. Programmers or analysts developed algorithms for solving specific problems, and data were formatted as required for each particular algorithm. As more and more processing is done on the same data, however, inordinate effort must be expended to recast the data in the format needed for each application. Furthermore, when new applications require a different representation of the data, old programs have to be modified to accept the new data formats.

It can be shown that, within certain constraints, information has an inherent structure. Information engineering is the branch of computer science dealing with the issues of data structure, the varying views which different programs take of this data, and its physical organization inside a computer. One of the results of this research is a set of programs called database managers.

Database management systems ideally decouple an application program from both the physical organization of the data and the validation process. Current commercially available systems fall short of this ideal, but offer a set of tools that allow considerable separation between application code and data management. For example, changes in the physical structure of the data can often be implemented in the application code by simple recompilation of the programs without any changes to the programs themselves.

The separation of application and data has other important consequences. One of the key problems in algorithm driven data processing is that the same data is duplicated in many different files. Keeping these files synchronized is a major difficulty. Database managers make it possible to have the data stored in only one place, but be accessible to all application programs. A number of formal technologies exist to develop databases that contain neither duplicate information nor "empty" records. Such "normalized" databases can also be shown to make most effective use of physical storage media. Furthermore, data base technology can, in a multi-user environment, enable several users to view data simultaneously and be simultaneously brought up to date.

Maintenance of existing programs and development of new modules in a large system is much less time consuming with a database approach. This makes code development less costly as well as quicker in elapsed time. Also, once the data has been captured in a database it is possible to perform ad hoc queries with relative ease. For information retrieval and decision support systems, such as ASAN, that must function in an unstructured evaluation and decision making and environment, these capabilities are essential.

Database systems may be characterized as predominantly high-volume data capture (transaction-oriented) systems, or predominantly analytic (management information and decision support) systems. The former support a highly structured work environment, requiring little or no user judgment; the latter support tasks that have relatively little inherent structure.

ASAN, intended to facilitate the conduct of environmental assessments, is in this latter category. The specific actions to be taken for an individual assessment vary widely from case to case. The data on which assessments must be based come from a variety of sources, and may need considerable reworking before they are in a form suitable for analysis. Furthermore, the state of the art in noise impact prediction is still evolving as more is learned about the effects of noise on people, animals, and structures.

Clearly, this calls for a highly modular system design, both with respect to the design of the application software and of the data structures. Modularity must be enforced throughout the system's design to fulfill the long-term objective of adaptability to changing needs at minimal cost. To achieve this, ASAN will consist of four largely independent structures:

1. Databases of generic and job specific information.
2. The User Interface, which controls the dialogue between the user and the system.
3. Analytic Models, implementing analysis and engineering "tools."
4. Control Logic -- the underlying software that binds the system together.

While total independence between hardware and software, among program logic modules, and between data and program structure are the goals of current system design approaches, practical limitations exist on the ability to implement these constructs on available hardware and software.

D.1 Role of Database Management in Current Design

As the design of the prototype system has evolved, it has become clear that it is desirable for ASAN to insulate the primary user from direct creation and manipulation of most of the databases it contains. This should reduce the end user's needs for training and computational skills, while encouraging concentration of effort on generation of environmental assessment work product. This design goal will be implemented by interposing a layer of control software between the database management software and the user. The control software will provide the user with a consistent, screen-oriented interface for all dealings with the system. Ideally, complete turnkey use of the software would not even require understanding the operating system of the host computer.

Furthermore, it has become evident that not all databases to be manipulated in ASAN are composed of text, and hence will not all be manipulable by conventional database management software. The databases to be incorporated in ASAN are listed elsewhere in this Report. Some of these databases are

graphic in nature, some numeric, and some textual. The database management software required to create and modify the various databases is discussed in separate sections below.

D.2 Nature of Database Management Packages

Before criteria for selecting a text database management system appropriate to the current use are identified below, it is important to understand that a database management system is essentially a set of procedures for creating and modifying files. Most such systems were originally written to deal with files of tabular data in the form of short text and numerical fields, especially business-oriented information such as personnel and inventory records. Such records are generally of a straightforward nature, typically containing employee data or part numbers that may be usefully associated with a variety of other columnar data (zip codes, dates, salary categories, etc.). Typical operations to be performed on tabular data include permuting (sorting, merging) and performing arithmetic functions upon rows and/or columns, and joining various subsets of rows and columns to make new tables.

The file management procedures that conventional database management packages provide are made available in a user-friendly manner through a "front end" (an interactive user interface or a batch mode report generator), or to other software through a "back end" (i.e., through direct subroutine calls). Since, in the present application, a layer of control software will be interposed between the user and the database management routines, there is no need for a front end of any sort. Likewise, certain other features of database management software packages are either irrelevant or undesirable for present purposes.

For example, "integrated" software packages which bundle together word processors, spreadsheets, graphics, and database management routines are counterproductive, since the superfluous capability is generally accompanied by size, complexity, speed and cost penalties. Such software is frequently optimized for the integrated application and is unsuitable as a generic building block for other applications.

It is also important for present purposes to distinguish between the database management software that is used to initially create the various text databases that will become part of the system, and the database management software that is to be incorporated the functional modules of ASAN.

The criteria for selecting the database management software to be used to create the databases initially are not the same as those for the database management software to be incorporated into the code. Because most database management software can "import" and "export" files either by converting idiosyncratic formats of other software or by accepting and producing standard ASCII strings, it is not essential that the same database management software system be used for both of the above purposes.

D.3 Selection Criteria for Text and Numeric Database Managers

The functional criteria for text and numeric database management within ASAN are described elsewhere in this Report. The most important criteria for selecting database management software for incorporation into ASAN are as follows:

- availability of a version that can execute in an IBM PC/AT MS-DOS/Zenith DOS operating environments.
- ability to rapidly and efficiently locate records.
- ability to accommodate variable length records.
- minimal consumption of system resources such as volatile and mass storage.
- absence of gratuitous capability, especially that which increases size or decreases performance.
- ability to operate while physically co-resident in main memory with other software, taking advantage of whatever remaining memory is available.
- imposition of minimal constraints such as arbitrary limits on number of bytes in a field, number of fields in a record, or number of records in a database.
- implementation of all capabilities as a library of functions callable from external C language programs.

The database management software market is a very fluid one, with vendors releasing upgrades and new products at frequent intervals. A review such as that provided in the next section therefore represents little more than a snapshot of the market at a particular point in time. Such a snapshot is unlikely to accurately reflect the availability and suitability of database management software at the time that production versions of ASAN are ready for distribution. A re-evaluation of commercial and public domain database management software packages available when ASAN is ready for release will clearly be needed, especially in the light of the advanced hardware capabilities expected several years hence.

D.4 Application of Criteria to Candidate Packages

A number of conventional database management software packages were evaluated for suitability for present purposes. These included Enable, Oracle, RS/1, dBase III+, BASIS, ETIS, and INFORMIX.

An initial screening of the different software packages revealed that several did not meet one or more of the above criteria. For example, BASIS, a Battelle-proprietary package, is software of considerable capability and complexity. While it has features that make it suitable for text analysis and litigation management applications and which might conceivably be of use for some applications in

ASAN, BASIS is not available for the Zenith-248 at the time of this writing, and has a host language interface only for FORTRAN, not for C.

Likewise, ETIS is VAX-based software that is unavailable for microcomputers. It does, however, offer dial-up access to a variety of databases, such as its CELDS subsystem, which could facilitate creation of a legislative and regulatory database for use within the ASAN software. This promise is, however, largely unfulfilled. An investigation of the contents of the legislative database for state-wide environmental regulations (intersecting "noise" with "Arizona") revealed only a single entry, for permissible automobile noise levels.

Enable is an integrated package that includes not only database management, but also business graphics and spreadsheet capabilities. These unwanted additional capabilities make the software cumbersome and slow. Enable also lacks a host language interface for C, and limits field sizes to 254 characters.

RS/1 is another integrated package that includes advanced statistical analysis and graphics capability. Although it is available in an IBM PC version, this version requires a minimum of 512 kB of RAM, and lacks a C-language interface. The greatest strengths of the IBM PC version of RS/1 are not in database management operations, but rather in spreadsheet-like data transformations, statistical inference, and technical graphics.

The ORACLE DataBase Management System is a conventional, tabular-oriented package that requires a minimum of 512 kB of RAM to execute in an IBM PC/AT environment. ORACLE also includes a report generator, a text formatter, on-line help, and extensive auditing and file security subsystems, as well as spreadsheet capability, business graphics, and forms management modules.

While Oracle has certain restrictions on table size (254 characters/field and 254 fields/table) it has no limitation on the number of tables. It is a full implementation of SQL, inherently slower than some partial implementations in execution speed. Any choice of a database management software package for ASAN involves a compromise between speed, feature set and memory requirements. The speed limitation of ORACLE is expected to be less of a consideration when faster processors become available for production versions of ASAN.

A version compatible with the Z-248 computer was announced shortly before the time of this writing. A unique feature of the ORACLE implementation for the Z-248 is that it operates in the Intel 80286 processor's protected mode; that is, in memory above 640 kB which is not used for other purposes by the operating system. Operation of the database management system in protected mode leaves more room for application code, an attractive feature.

The dBase III+, a popular database management system, was written originally for eight bit microcomputers. It lacks certain of the more sophisticated features of database management packages

which have migrated to the PC world from mainframe versions. Although it is callable in assembly language, it lacks a C language interface. It also imposes several constraints on numbers of fields per record (128) and field lengths (fewer than 254 characters). It is, however, relatively quick in locating records in large databases. Its retail price is approximately $700.

INFORMIX-SQL is a database management package that has little excess capability, executes in the desired host computer environment, is callable from C language programs, and is both compact and fast. It is not, however, a complete implementation of SQL. While it will accommodate field sizes of up to 32,768 characters, and imposes no record number limitations, it shares with most other microcomputer systems a limit on the number of data tables that can be open at any given time.

INFORMIX-SQL appears to be one of the fastest packages available. A test of performance of retrieval and display of indexed and non-indexed records performed by PC Magazine (June, 1986) showed INFORMIX to be the fastest system of all of those identified above. For example, to retrieve non-indexed records in this benchmark, it took Enable 11.5 seconds, Oracle 7.0 seconds, dBase III+ 3.7 seconds, and INFORMIX only 2.5 seconds. The PC version of INFORMIX retails for about $120.

D.5 Recommendation for Text Database Management

D.5.1 Text Data Entry

The preferred database management system for the initial creation and entry of records into ASAN prototype system databases is dBASE III+. Its principal advantages are that it is widely available and well understood by many, relatively convenient to customize for use in data entry, and sufficiently complex to accommodate a fairly involved database schema. The dBASE III+ can also output pure ASCII files readable by other database management software systems; it is not, however, the software system of choice for incorporation into the executable code of ASAN.

D.5.2 Run Time Text Data Manipulation

The preferred database management system for manipulating tabular and text database files in the prototype version of ASAN is ORACLE. Its major advantages for this purpose include its ability to operate in the ASAN host computer's protected mode, the ease of C language access, and its sophisticated auditing capabilities. Its major disadvantage is that it is somewhat slower than alternative SQL-based database management software.

The choice is a compromise. On the one hand, it is essential that the database manager used in

ASAN have the data file integrity mechanism that comes with sophisticated mainframe-bred systems. On the other, these kinds of systems tend to require large amounts of memory. The operating systems that will make larger memory available will not be released in time for the demonstration software, nor will the microprocessors (e.g., the Intel 80386) that can implement these systems efficiently. As long as the database management software incorporated into the production version of ASAN is SQL-based, however, there should be minimal difficulty in transitioning from the prototype to the production versions.

D.6 Geodatabase Management

D.6.1 Comparison of Geodatabase Systems

Parker, Limp, Farley, and Johnson (1986) have reviewed the capabilities of the principal geoinformation systems. The following discussion highlighting the differences among several currently available proprietary and public domain systems is excerpted in part from Parker et al.

ARC/INFO: This is perhaps the premiere proprietary system. It was originally designed for Data General and PR1ME 32-bit minicomputers, but portions have been ported to the PC environment. ARC/INFO processes vector and raster data in a very efficient manner, minimizing data redundancy by producing compact data structures. It can accept and convert data from other cartographic systems, including non-topological data from scanners, coordinate data from photogrammetric work stations, Landsat satellite data, U.S.G.S. digital line graph formats, and many other types of data. ARC/INFO is widely used for a variety of land use management tasks. Mainframe and minicomputer versions of ARC/INFO cost as much as $90,000; PC versions of reduced capability cost roughly $10,000. Its principal disadvantages for present purposes are its cost, its reduced capability in PC form, the lack of availability and the proprietary nature of its source code.

ERDAS (Earth Resources Data Analysis System) is an image processing and computer mapping system. Although it can merge, store, and display a variety of digital data, it lacks the primary analysis capabilities essential to a comprehensive geoinformation system.

MOSS (Map Overlay and Statistical System) is a Fortran-based, batch mode set of programs for encoding, transformatting, analyzing, and displaying map and other geo-based information. Development of MOSS was originally sponsored in the early 1970s by a consortium of several federal land use management agencies. No single organization currently has full responsibility for maintaining and developing MOSS. This decade-old software is composed of three major subsystems: Analytical Mapping System (AMS) for digital data entry, MAPS for data processing, analysis, and display, and Cartographic Output System (COS) for enhanced cartographic plotting. Map data may be manipulated in

vector, cell, or raster formats. The choice of data formats depends on the eventual use of the data. In general, vector maps are preferred for high precision cartographic analysis, whereas cell maps are preferred for complex modeling procedures.

MOSS runs on 16 bit Data General minicomputers. In addition to its unavailability for ASAN's target host computer, its disadvantages include its complex nature, diffuse support and uncertain configuration control, and batch mode orientation. Porting MOSS from minicomputer, batch mode Fortran to an interactive, personal computer C language environment would be a very expensive proposition.

GRASS is a public domain software system written in the C language at the U.S. Army Construction Engineering Research Laboratories. Version 2.0 of the system has just been released, while additional development efforts continue. GRASS is written as a modular library of procedures to facilitate porting to different host computers.

GRASS is a grid cell (raster) based system with a capacity for rapid production of extensive map comparisons and overlays. It was designed ab initio as a modern interactive system, and performs its operations considerably faster than other geoinformation systems. It accepts digital geodata in a variety of formats, and can re-sample the original data to change scales efficiently. GRASS stores data in a compressed form to minimize mass storage requirements for geodatabases.

D.6.2 Recommendations for Geodatabase Management Software

The preferred geodatabase management system is GRASS. BBN has received assurances of support from the CERL development team in developing the port to the MS/DOS environment, including training and assistance in re-configuring the source code. The quid pro quo for this valuable assistance is access to any further public domain code development of GRASS in the PC environment.

GRASS is preferred to other geodatabase systems for the following reasons among others:

- The original software development group is intact and actively working on enhancements to the system. Code development is proceeding in a systematic and rigorous fashion, in a modern development environment. A single agency (U.S. Army Construction Engineering Research Laboratory) has complete authority and responsibility for the development of GRASS, and is providing excellent documentation and support for the package.
- GRASS uses system resources efficiently, especially mass storage and graphic display hardware subsystems.
- GRASS produces well composed and formatted graphic output, in color, on a variety of display and hardcopy devices.
- GRASS is structured as a library of independent subprograms, so that unnecessary functionality can be easily omitted.

- Source code for GRASS is available at negligible cost.

Appendix E
Description of Screen-Oriented User Interface

E.1 Overview

The U is a BBN-proprietary set of tools for building and using a screen-oriented interactive interface to user-written application software. U runs under the Unix operating system and look-alikes (e.g., Xenix).

The interface is driven by a text file called a Screen Description File (SDF). At run time, the interface controls all interaction with the application according to the directives in the SDF.

Programmers use U by preparing a SDF. A parser analyzes the SDF to generate several files which can be compiled and linked with application software to form a complete executable program.

The interface is terminal independent: it will run on any terminal so long as the operating system knows what type of terminal it is.

The interface can accept commands from console pointing devices in a device independent manner, so that one console pointing device can be substituted for another, or so that the interface can operate without any pointing device if need be.

E.2 SDF Building Blocks Summary

The interface manages the actual terminal screen. The application code must NEVER write to the terminal screen directly.

"Building blocks" are the pieces which constitute the display, such as "windows" and "buttons."

A "screen" is a stored representation of how the actual terminal screen might look. One screen is always displayed. Screens are made out of "windows." A "window" is a portion (all or part) of a screen. Windows are made out of "datums," "text lines," "text blocks," "buttons," "drawn lines," and "drawn boxes."

A "datum" is a description of a single variable, with the information necessary to display and manipulate it. A "variable" is an external variable in your application software. A "text line" is a single line of text. A "text block" is any amount of text, in a file. A "button," when pushed, causes something to happen.

E.3 Selected Details

The Terminal Screen is assumed to be 24 rows by 80 columns. The bottom row is reserved for status messages. The "screen" size is therefore 23 rows by 80 columns. By pressing mouse buttons or keyboard keys, the user can move the screen cursor. The cursor can ONLY rest on buttons, datums and text blocks. When the cursor rests on an item, that item is "picked." Exactly one item is "picked" at all times. The status line always shows the mouse button / keyboard key assignments which are meaningful for the "picked" item. When a text block is "picked," it can be "scrolled."

Datums and buttons are normally pickable, but can be made unpickable if so desired. Pressing the assigned mouse button or keyboard key "activates" the "picked" item. When a datum is "activated," it can be "updated" (its value changed by typing a new value) and its associated "actions" executed. When a button is "activated," its associated "actions" are executed.

When the end user moves the cursor, the interface decides its next position unless provision is made for the application code to do so. "Actions" are things to do when a datum or button is "activated." Any number of actions can be specified for an item. Actions are executed sequentially, in the order of their appearance in the item's action list. "Update" (== "accept a new data value") is the default first action for datums. Buttons ALWAYS have explicit action lists. Many actions are built into the interface.

The user can obtain help at any time, by pressing the assigned "help" keyboard key or mouse button. "Help" messages are a form of "pop-up window": they appear when invoked, can be "scrolled" up and down, and disappear when the cursor is moved to another item or screen.

E.4 Descriptions of ASAN User Interface Screens

E.4.1 Introductory Screen

The first screen that users will see upon starting the program will announce the program name and release version, and log the user into ASAN. The login is intended primarily for audit trail purposes, although it will also permit rudimentary data security through user recognition. Once successfully logged in, all users will be accorded equal privilege to read and write any user-accessible data. Although ASAN may protect certain critical data from deletion, it will not bar knowledgeable users from entering other data. Information that a local user need not manipulate, and data too sensitive for access by local users, will not be made available in the personal computer-based system.

E.4.2 Top Level Control Menu Screen

The first screen seen after a successful login will permit the user to choose one of five major actions:

- Problem Definition

  Selection of this option will call up a separate screen (v.i.) that will permit the user to enter (update or modify) or delete information in one of the databases.

- Data Analysis

  Selection of this option will call up a separate screen (v.i.) that will permit the user to exercise the various exposure estimation and effects determination modules.

- Report Generation

  Selection of this option will call up a separate screen (v.i.) that will permit the user to create text and graphic output products.

- Housekeeping

  Selection of this option will call up a separate screen (v.i.) that will permit the user to perform a variety of system maintenance functions.

- Tutorial

  Selection of this option will call up a separate screen that will provide access to a self-instructional mode.

The Top Level Control Menu screen, as well as all subsequent ones, will also contain provisions for users to "undo" the last action taken, to reset ASAN and its data structures to a previous state, and to access help files.

E.4.3 Problem Definition Screen

This screen provides the user with access to the portions of ASAN that deal with the problem definition phase of an environmental assessment. The screen and any associated subscreens will permit the user to select one or more databases and the operations (addition, deletion, modification, composition of maps) to be performed on them.

E.4.4 Data Analysis Screen

This screen will provide users with access to the following software capabilities:

- identification of one or more noise sources and affected points or areas on a map display.
- performance of a standard, quick-look analysis for the selected noise sources and affected points or areas.
- estimation of exposure or determination of any combination of noise effects to specified levels of detail, on demand.
- comparison of alternative combinations of noise exposure and noise effects.

E.4.5 Non-Noise Effects Screen

This screen provides access to potential non-noise related extensions of ASAN.

E.4.6 Effects Comparison Screen

This screen permits users to compare the impacts of named alternative exposure conditions on affected areas. Functions accessible from this screen include listing, storage, and retrieval of named sets of exposure conditions and affected areas, and rank ordering of alternatives by environmental impacts.

E.4.7 Report Generation Screen

This screen will permit users to produce a variety of standard output products, or to combine portions of pre-formatted text and graphics into customized reports.

E.4.8 Housekeeping Screen

This screen will permit users to command the software to perform a variety of functions associated with routine system maintenance such as backing up, restoring and purging data.

END

DATE

FILMED

DTIC

10-88